BEFORE THE UNITED STATES PATENT AND TRADEMARK OFFICE
BOARD OF APPEALS AND INTERFERENCES

In re Application of	:	Tribble, Rob
Serial No.	:	09/668,125
Filed	:	9/21/2000
Art Unit	:	3626
Confirmation No.	:	1382
Examiner	:	Frenel, Vanel
Title	:	Business Rules System
Atty. Docket No.	:	NETS0044

Date: June 06, 2007

Honorable Commissioner of Patents and Trademarks
P O Box 1450
Alexandria, VA 22313-1450

AMENDED BRIEF ON APPEAL

This is amended appeal is in response to the Notification of Non-Compliant Appeal Brief dated 04-04-06.

Per Appeal Brief filed 01-13-06, the Primary Examiner of Group Art 3626 refusing Claims1-4, 6-11, 13-18, 20, and 21 set forth in Appendix A hereto. The application figures appear in Appendix B.

TABLE OF CONTENTS

REAL PARTY IN INTEREST 4

RELATED APPEALS AND INTERFERENCES 4

STATUS OF CLAIMS 4

STATUS OF AMENDMENTS 4

SUMMARY OF CLAIMED SUBJECT MATTER 4

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL 8

ARGUMENT AND DISCUSSION 8

CONCLUSION 25

APPENDIX A - Applicant's Claims

APPENDIX B - Applicant's Figures

APPENDIX C – Evidence

APPENDIX D – Related appeals and interferences

APPENDIX E - Non-Final Office Action, dated 8/27/04

APPENDIX F - Final Office Action, dated 2/10/05

APPENDIX G - U.S. Patent No. 5,400,248

APPENDIX H - U.S. Patent No. 4,210,962

REAL PARTY IN INTEREST

The subject application was assigned to American Online, Inc., a corporation of the state of Delaware. The assignment to was recorded in the U.S. Patent & Trademark Office on the date of 2/6/01, at Reel 11532, Frame 238.

EVIDENCE

No evidence is submitted herewith (see Appendix "C").

RELATED APPEALS AND INTERFERENCES

No other appeals or interferences are known to be related to the subject patent application (see Appendix "D").

STATUS OF CLAIMS

The application was originally filed with Claims 1-21. Claims 5, 12, and 19 have been canceled. Therefore, Claims 1-4, 6-11, 13-18, 20, and 21 are pending in the application.

Claims 1-4, 6-11, 13-18, 20, and 21 stand rejected; these claims are being appealed.

STATUS OF AMENDMENTS

Amendments were submitted 2/10/04, 7/7/04, and 11/29/04, and have been entered by the Examiner.

SUMMARY OF CLAIMED SUBJECT MATTER

The invention provides a business rules system. The system creates dynamic solution sets in response to interactions between real-world objects. In addition, the invention provides a system that allows the user to easily create and maintain the rules describing

and governing a business system.

A preferred embodiment of the invention creates a dynamic solution set between objects that have never been associated with one another prior to that particular instance of evaluation. A rule engine evaluates the associated objects. The rule engine is programmed to evaluate a complex situation, come up with a solution and, as a result, not require the intervention of a human being.

The invention's rule engine enables the user to customize the behavior of business objects (*e.g.*, access control, order management, catalogs, and membership) to meet sophisticated business requirements. The invention provides a graphical user interface that allows a user to rapidly set up and maintain Business Rules in real time - with no source code recompilation needed. Domain tables are used to define the overall set of possible values for a given attribute of an object.

Each rule may have one to many classes of interacting objects. A voter is a member or business object such as a product, price list, or ship-to location that provides input to be evaluated by the rule. Each instance of a voter may have a data value (vote) for a specific rule. Voters also have a hierarchical inheritance.

Rule resolution strategies are used by the invention to resolve any conflicts between specific preferences of voters and to determine the correct answer to be used in an application program. The sequence of business objects and the order of the values included in each object determine the values of the results. The rule resolution logic returns only one answer for a question that references a set of business objects.

The following presents a summary of the claimed matter, referring to the specification by page and line number and to the drawings, as appropriate, by reference characters. Additionally, in Claims 4,8,11,15, and 18 every mean plus function element (35 USC 112 ¶6) is identified, and the structure, materials, or acts described in the specification as corresponding to each claimed function is set forth with reference to the specification by date and line number and to the drawing, as appropriate, by reference characters:

1. A process for evaluating business objects (page 6, lines 11-14; page 9, lines 10-20) with no prior association and creating dynamic solution sets based on said evaluation in a computer environment, comprising the steps of:

providing a rule engine(page 6, lines 6-9; page 23, lines 8-20; Fig. 9:904; Fig. 10:105)

wherein said rule engine evaluates said business object (page 6, lines 6-10; Fig. 1:103);

providing administration means for allowing a user to maintain preferences for a specific instance of a business object(page 6, lines 11-14; page 11, lines 8-13; page 12, lines 7-23; Fig. 7:701-703; Fig. 8: 801-803);

wherein said rule engine uses business rules (page 10, lines 10-23) to evaluate a relationship between said business objects;

wherein each business object is a voter (page 14, lines 8-24) that provides votes that are evaluated by said business rules;

providing rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set(page 16, line 31 to page 18, line 23; Fig. 5A and 5B: 502, 505-516); and

wherein a sequence of voters and an order of the votes included for each voter determine values in said solution set (page 19, lines 19-22).

4. The process of claim 1, further comprising the step of:

providing rule administration means for allowing a user to define business rule attributes(page 6, lines 11-14; page 11, lines 8-13; page 12, lines 7-23; Fig. 7:701-703; Fig. 8: 801-803);.

8. An apparatus for evaluating business objects(page 6, lines 11-14; page 9, lines 10-20) with no prior association and creating dynamic solution sets based on said evaluation in a computer environment, comprising:

a rule engine(page 6, lines 6-9; page 23, lines 8-20; Fig. 9:904; Fig. 10:105);

wherein said rule engine evaluates said business objects(page 6, lines 6-10; Fig. 1:103);

administration means for allowing a user to maintain preferences for a specific instance of a business object (page 6, lines 11-14; page 11, lines 8-13; page 12, lines 7-23; Fig. 7:701-703; Fig. 8: 801-803);

wherein said rule engine uses business rules to evaluate a relationship between said business objects(page 10, lines 10-23);

wherein each business object is a voter that provides votes that are evaluated by said business rules(page 14, lines 8-24);

rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set(page 16, line 31 to page 18, line 23; Fig. 5A and 5B: 502, 505-516); and

wherein the sequence of voters and the order of the votes included for each voter determines values in said solution set(page 19, lines 19-22).

11. The apparatus of claim 8, further comprising:

rule administration means for allowing a user to define business rule attributes(page 6, lines 11-14; page 11, lines 8-13; page 12, lines 7-23; Fig. 7:701-703; Fig. 8: 801-803).

15. A program storage medium readable by a computer, tangibly embodying a program of instructions executable by the computer to perform method steps for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment, comprising the steps of:

providing a rule engine

wherein said rule engine evaluates said business objects;

providing administration means for allowing a user to maintain preferences for a specific instance of a business object;

wherein said rule engine uses business rules to evaluate a relationship between said business objects;

wherein each business object is a voter that provides votes that are evaluated by said business rules;

providing rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set; and

wherein the sequence of voters and the order of the votes included for each voter determines values in said solution set.

18. The method of claim 15, further comprising the step of:

providing rule administration means for allowing a user to define business rule attributes(page 6, lines 11-14; page 11, lines 8-13; page 12, lines 7-23; Fig. 7:701-703; Fig. 8: 801-803);.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-4, 6-11, 13-18, 20, and 21 were rejected under 35 USC 103 as being unpatentable over USPN 5,400,248 (Chisholm; Appendix G) in view of USPN 4,210,962 (Marsh; Appendix H).

The foregoing rejections were made in the non-final Office Action, dated 8/27/04, which is enclosed under Appendix E. These rejections have been maintained in the final Office Action, dated 2/10/05 (Appendix F).

ARGUMENT AND DISCUSSION

Subparagraph (i) B 35 USC 112, FIRST PARAGRAPH

No grounds of rejection exist under this subparagraph.

Subparagraph (ii) -- 35 USC 112, SECOND PARAGRAPH

No grounds of rejection exist under this subparagraph.

Subparagraph (iii) -- 35 USC 102

No grounds of rejection exist under this subparagraph.

Subparagraph (iv) -- 35 USC 103 REJECTIONS

The Office Action dated 8/27/04 rejected Claims 1-4, 6-11, 13-18, 20, and 21 under 35 USC 103 as being unpatentable as follows:

Claims 1-4, 6-11, 13-18, 20, and 21 were rejected under 35 USC 103 as being unpatentable over USPN 5,400,248 (Chisholm) in view of USPN 4,210,962 (Marsh).

Applicants previously traversed, and presently appeal these rejections. The Office Action of 2/10/05 maintained the previous grounds of rejection. The claims are patentable as-is, because the Examiner has failed to satisfy the requirements to establish a *prima facie* case of obviousness, as discussed in greater detail below.[1]

First, the *prima facie* obviousness case is incomplete because, even if the references were to be combined as suggested (albeit improperly, as discussed below), the combination still does not teach or suggest all the claim limitations.[2] To support the conclusion that the claimed invention is directed to obvious subject matter, either the references must expressly or impliedly suggest the claimed invention or the Examiner must present a convincing line of reasoning as to why the artisan would have found the claimed invention to have been obvious in light of the teachings of the references.[3]

All words in a claim must be considered in judging the patentability of that claim against the prior art.[4] In the following, Applicant will address the claims that are rejected by the Examiner. It is respectfully submitted that these claims are not obvious to a person skilled

---

1 MPEP 2142.

2 MPEP 2142, 2143.03.

3 *Ex Parte Clapp*, 227 USPQ 972, 973 (Bd. Pat. App. & Inter. 1985). MPEP 706.02(j).

4 *In re Wilson*, 424 F.2d 1382, 165 USPQ 494, 496 (CCPA 1970). MPEP 2143.03.

in the art when considering the Examiner's proposed combinations.

Claim 1

The Examiner has stated (Office Action, dated 8/27/04), with regard to Claim 1, Chisholm discloses:

providing a rules engine;

(col. 5, lines 17-35: the Examiner interprets Chisholm's voter administrator program to be a form of "rules engine": in the Office Action of 2/10/05, the Examiner further stated that he has "relied upon Chisholm for the teaching of "each business object being a voter (See Chisholm, col. 5, lines 12-34) that provides votes that are evaluated by the business rules, wherein a sequence of voters and an order of votes determines values in a solution set (See Chisholm, col. 9, lines 35-50; col. 10, line 55-68 to col. 11, line 3) which correspond to Applicant (*sic*) claimed feature." As discussed below, this is not what Chisholm teaches.)

wherein said rules engine evaluates said business objects;

(col. 5, lines 23-34: the Examiner interprets "votes" as "business objects." Again, as discussed below, this is a mischaracterization of "votes" as taught by Chisholm and "business objects" as taught by Applicant. They are in no way analogous.)

providing administration means for allowing a user to maintain preferences for a specific instance of a business object;

(col. 5, lines 12-34)

wherein said rule engine uses business rules to evaluate a relationship between said business objects;

(col. 5, lines 55-68: the Examiner understands conditional votes are votes having a

relationship to other votes. Again, as discussed below, the Examiner is clearly mistaken.)

> wherein each business object is a voter that provides votes that are evaluated by said business rules;

(col. 5, lines 12-34)

> providing rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set; and

(col. 5, lines 55-65; col. 8, lines 10-44)

> wherein a sequence of voters and an order of the votes included for each voter determine values in said solution set.

(col. 9, lines 35-50; col. 10, lines 55-68 to col. 11, line 3)

The Examiner acknowledges that Chisholm does not "disclose a process for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment." Nonetheless, the Examiner finds this aspect of the invention set forth in Claim in Marsh (col. 3, lines 8-11; col. 4, lines 22-28).

With regard to Claim 1, the proposed combination of references fails to teach the claimed invention. Looking at the cited references, Chisholm discloses a voting system that allows voters to express and cast votes that are conditional on the votes of others of a voting group. Chisholm fails to teach or suggest providing a rule engine, which evaluates business objects with no prior association and uses business rules to evaluate a relationship between the business objects, each business object being a voter that provides votes that are evaluated by the business rules, wherein a sequence of voters and an order of the votes determine values in a solution set, as claimed in independent Claim 1.

Marsh does not remedy any of the deficiencies of Chisholm. Marsh discloses a parallel/pipeline processor designed to rapidly solve optimization problems with dynamic programming. Marsh fails to teach or suggest providing a rule engine, which evaluates business objects with no prior association and uses business rules to evaluate a relationship between the business objects, each business object being a voter that provides votes that are evaluated by the business rules, wherein a sequence of voters and an order of the votes determine values in a solution set, as claimed in independent Claim 1.

Furthermore, Chisholm fails to teach or suggest a combination with Marsh and Marsh fails to teach or suggest a combination with Chisholm. Chisholm discloses a voting system for casting and tabulating votes and displaying the results, while Marsh is concerned with processors having parallel pipeline architecture. It would be impermissible hindsight based on Applicant's own disclosure to incorporate the teachings of Marsh into Chisholm. Moreover, such a combination would still fail to teach or suggest providing a rule engine, which evaluates business objects with no prior association and uses business rules to evaluate a relationship between the business objects, each business object being a voter that provides votes that are evaluated by the business rules, wherein a sequence of voters and an order of the votes determine values in a solution set, as claimed in independent Claim 1.

The following sets forth those sections of Chisholm and Marsh that are relied upon by the Examiner in rejecting the claimed invention, with Applicant's comments in [brackets]:

Chisholm

col. 5, lines 12-34:

> FIG. 3 is a flowchart of the preferred embodiment of the invention. In this case voting begins when an individual, the proposal originator, develops one or more vote proposals. A vote proposal may take many forms. It may be able to be voted on

affirmatively or negatively, or it may contain multiple alternatives that can be prioritized, that is, ranked, by voters. A vote administrator is a person or program charged with specifying terms and conditions of a voting. The vote administrator may be the same as, or different from, the proposal originator. Either the proposal originator or the vote administrator must enter the proposal into the system in electronic form (step A). In the preferred embodiment the proposal is entered by keyboard. If the proposal is entered by the originator, the system makes it available electronically to the vote administrator, for example on a computer screen. Either before or after proposals are submitted, the vote administrator specifies the terms and conditions for the votes (step B), such as who may vote, voting deadline(s), and constraints, if any, on allowed vote types. The voting system then notifies members of the group through the voting units or through other means that there are one or more proposals to be voted on (step C).

[Comment: this teaching concerns a human-mediated system the requires a human vote administrator and a set of human voters. In Claim 1, a "rules engine" performs the function of evaluating "business objects." The rules engine is not a human, nor are Applicant's business objects humans.]

col. 5, lines 55-68:

The system then processes the votes to compute their values (represented as signals). Depending upon the types of votes allowed by the vote terms (specified by the vote administrator) and upon the specific votes cast by the voters, the processing performed will vary. As shall be seen below, a processed vote may have either a unique computed value, multiple values, or no meaningful value (i.e., no solution). An unconditional vote always has a unique value--either yes, no or abstain--but a conditional vote may have either a unique value, multiple values, or no meaningful value. The vote terms determine, among other things, how multiple values and no meaningful values of votes are handled. For example, if the computed value of a vote is either yes or no, the terms may specify that "yes" will always be selected and presented as output. This approach can help build

consensus among the voters. Or, the terms may specify that both values must be presented as output. If a vote has no meaningful computed value, the terms may specify that this fact be presented as output, or they may specify that the voter who casts that vote change his or her vote. Votes that have multiple computed values are called herein indeterminate. Votes that have no meaningful values or solutions are called herein unresolvable.

[Comment: this passage teaches tabulation of human-entered votes: it has absolutely nothing to do with a computer evaluating a relationship between business objects.]

col. 8, lines 10-44:

The terms set by the vote administrator determine whether the system presents or reports all or a subset of the multiple solutions, when they arise. In the interest of consensus, the default assumption where a group of votes has multiple solutions is usually the one with the most yeses.

The vote administrator may specify any of the following output alternatives: i) present all solutions; ii) present all solutions that meet certain criteria, such as all solutions with three or more yeses; or iii) present only those solutions with either the most yeses or the most nos; or iv) present an "average" of all solutions. In addition to any of these four alternatives, the system can recommend to voters whose votes cause the multiple solutions how their votes can be modified to eliminate multiple solutions.

A different problem is encountered in the following scenario:

Voter #1 votes yes if #2 votes yes; else, no.

Voter #2 votes no if #1 votes yes; else, yes.

Restating this example more simply, A votes the same as B, and B votes the opposite of A. There is no solution. Allowing voters to vote opposite the way of others can lead

to this result. These votes are called unresolvable.

In these cases, the system reports whose votes contain no solution, either to only the individuals casting those votes or to the group as a whole, depending upon the terms of the vote. One or both of the voters need to change their votes to make a solution possible. A partial solution is a subset of all of the votes that have a solution. When there are unresolvable votes, the voting system identifies the partial solutions with the most votes, and identifies the unresolvable votes.

[Comment: Applicant has used the terminology "voter" and "vote" to refer to a weighting given the business objects. The Examiner has conflated the Applicant's meaning with that of a human voter. In Chisholm, human voters vote; in the claimed invention, business objects are weighted to resolve conflicts in determining a solution set for a business problem. In this regard, a few definitions from Applicant's specification will contrast the invention with Chisholm:

What Are Rules?

Business rules are configurable, generalized statements of how common processing methods are applied to a specific intersection of data. Business rules determine:

- privileges - what a user can create, view, update, select, or delete.
- application of business processes - what processes are accepted, such as buying with credit cards.
- business relationships - relationships between selling and buying companies, *e.g.*, what price list is used when Catalog X is selected or where the item is shipped to or from.
- choices and default values - the defaults for options such as ship-to address or payment methods. (page 10, lines 10-23)

Rule Components

The main components of a rule are:

- voters
- votes
- resolution strategy

A voter is a member or business object such as a product, price list, or ship-to location that provides input to be evaluated by the rule. This input is known as votes. The voters have a specified sequence within a rule. This sequence determines the order in which a particular voter's votes are evaluated by the resolution strategy assigned to the rule.

Voters can be data objects, domain tables, or the results of other rules. For example, a voter represents a specific instance of a object that is related to the object (*e.g.*, an order) being operated on by the calling program. (page 14, lines 10-24; emphasis added: note that "voters" as used by Applicant are not humans casting votes on issues, as taught by Chisholm)

Chisholm, col. 9, lines 35-50:

The consensus building chart allows a group to see how close or far away it is from achieving consensus, or from achieving a coalition of a particular size. If a voting has successive iterations, the graph may vary with each iteration. In that case, the graph can be updated or played back in real time, allowing voters to review an animated history of the group's preferences as they have evolved, to visually gauge the momentum towards consensus, or to pinpoint turning points or major events in the group's dynamics. FIG. 5 illustrates a consensus building chart with three successive iterations; y.sub.1, y.sub.2, and y.sub.3. The figure shows how the acceptance of the proposal has changed with each successive vote, indicating that at least some of the individual voters have modified their votes. Similarly, if y.sub.1,

> y.sub.2, and y.sub.3 were to indicate three different proposals, then a chart looking like FIG. 5 could be used to gauge the relative acceptance of each distinct proposal.

[Comment: this passage refers to building a consensus among humans, *i.e.* the voters. The invention concerns business objects, not humans. The invention does not seek to build a consensus among humans, it determines a solution set for a business problem.]

col. 10, lines 55-68 to col. 11, line 3:

> At the beginning of stage 1 (step H), the system examines the votes in whatever order they happen to be in, and identifies the unconditional ones (yes, no, abstain, no-vote). As the system examines each vote, it also evaluates any conditional votes that have become determinable as a result of unconditional votes now determined. Votes so determined are conditional votes that are dependent only on the unconditional votes. Then the system passes through the list again and evaluates all new votes that are dependent only on the ones previously determined, either conditional or unconditional. This process is repeated until an iteration occurs on which no new votes are determined. If all votes in the group have been determined by this process (step I), the system is finished and the results are displayed. If all votes have not been determined, we proceed to stage 2 (step J in FIG. 9A; P in FIG. 9B).

[Comment: Again, the votes being examined are votes that were cast by humans. This passage merely teaches the tabulation of votes that are human cast. It has nothing to do with votes associated with business objects. The contrast is quite striking – the invention uses the notion of a vote as a weighting for business objects; Chisholm merely uses votes in the conventional, human cast sense. Further, this passage has nothing to do with evaluating the votes in such manner that the sequence and order determines a solution set value. Chisholm is not interested in determining a solution set value, but only with tabulating human cast votes. Further, this passage does not teach or suggest that both the sequence and order in which votes are cast could affect an outcome. Rather, Chisholm only teaches a mechanism for resolving conditional votes. In fact, Chisholm teaches away

from determining a solution set value based upon vote sequence and order by stating that "the system examines the votes in whatever order they happen to be in..."]

Marsh

col. 3, lines 8-11:

> As mentioned previously, dynamic programming is an approach for solving optimization (maximization or minimization) problems, relying on dissecting the main optimization problem into many intermediate optimization problems.

[Comment: What is the person skilled in the art to learn from this passage? The Examiner relies upon this passage, and the next, to teach "a process for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment"; and to provide a motivation for combining this teaching with Chisholm. If the inventors of the claimed invention were to give Marsh and Chisholm to a programmer, it would not be possible to build the claimed invention without providing inventive effort.]

col. 4, lines 22-28:

> Moreover, most dynamic program solutions specify the optimum transition at each state of the system for every stage; thus, a dynamic programming solution can be implemented as a feedback (closed-loop) controller, in which the state of the system is constantly measured and the corresponding optimum control is applied.

[Comment: ????]

col. 2, lines 21-25:

> The present invention provides a processor especially designed to solve dynamic programs in a minimum of time and without the necessity of complex interprocessor

communication.

[Comment: The Examiner relies upon this passage to teach the person skilled in the art that there is a motivation to solve a dynamic program with a minimum amount of time and without complex interprocessor communication. How? Marsh merely teaches a processor architecture. The Examiner is suggesting that anyone aware of Marsh's processor could adapt Chisholm's human vote tabulation system to produce the claimed business object evaluation system.]

A reference itself must sufficiently describe the claimed invention to have placed the public in possession of it.[5] Even if a claimed invention is disclosed in a printed publication, that disclosure will not suffice as prior art if it was not enabling.[6] The lack of teaching in both Chisholm and Marsh of Applicant's claimed inventive elements of:

> "a rule engine... [that] evaluates ... business objects; ...administration means for allowing a user to maintain preferences for a specific instance of a business object; ... [the] rule engine ... [using] business rules to evaluate a relationship between ... [the] business objects; wherein each business object is a voter that provides votes that are evaluated by ... [the] business rules; ... rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set; ... wherein a sequence of voters and an order of the votes included for each voter determine values in ... [the] solution set"

means that, not only would the skilled person be required to supply the missing elements, but that the skilled person would have to determine how to implement these elements as well. Accordingly, the Examiner's combination is fatally defective in at least two regards.

In addition to the reasons given above, the *prima facie* obviousness case is also

---

5 *Paperless Accounting, Inc. v. Bay Area Rapid Transit System*, 231 USPQ 649, 653 (Fed. Cir. 1986). *Ex parte Gould*, 231 USPQ 421 (CCPA 1973).

6 *Id.*

defective because there has been no suggestion or motivation, either in the references themselves, or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings.[7]

The law requires clear and particular evidence of a suggestion, teaching, or motivation to combine references or modify reference teachings[8] Broad conclusory statements regarding the teaching of multiple references, standing alone, are not "evidence."[9] In addition to demonstrating the propriety of an obviousness analysis, the Federal Circuit recognizes the value of particular factual findings regarding the suggestion, teaching, or motivation to combine because this serves a number of important purposes, including: (1) clear explication of the position adopted by the Examiner and the Board; (2) identification of the factual disputes, if any, between the applicant and the Board, and (3) facilitation of review on appeal.

Accordingly, the *prima facie* case of obviousness is lacking because there has been no showing of the legally required suggestion or motivation to modify the reference or to combine reference teachings.

Rather than a legally sufficient suggestion or motivation, modification of Chisholm to incorporate features from Marsh is simply a result of hindsight reconstruction. This amounts to simply hunting for a missing feature until it is found in some secondary reference, and then reflexively pairing this reference with the primary reference. However, it is improper to attempt to establish obviousness by using the Applicant's specification as a guide to combining different prior art references to achieve the results of the claimed invention.[10] The teaching or suggestion to make the claimed combination must be found in

---

7 MPEP 2142.

8 See, e.g., *C.R. Bard, Inc. v. M3 Sys., Inc.*, 48 USPQ2d 1225, 11232 (Fed. Cir. 1998).

9 See, e.g., *C.R. Bard, Inc. v. M3 Sys., Inc.*, 48 USPQ2d 1225, 11232 (Fed. Cir. 1998)

10 *Orthopedic Equipment Co., Inc. v. United States*, 702 F.2d 1005, 1012, 217 USPQ 193, 199 (Fed. Cir. 1983).

the prior art, and not based on applicant's disclosure.[11] The critical inquiry is whether there is something in the prior art as a whole to suggest the desirability, and thus the obviousness, of making the combination.[12] Obviousness is tested by what the combined teachings of the references would have suggested to those of ordinary skill in the art.[13] But it Acannot be established by combining the teachings of the prior art to produce the claimed invention, absent some teaching or suggestion supporting the combination.[14] And teachings of references can be combined only if there is some suggestion of incentive to do so.[15]

To imbue one of ordinary skill in the art with knowledge of the invention in suit, when no prior art reference or references of record convey or suggest that knowledge, is to fall victim to the insidious effect of a hindsight syndrome wherein that which only the inventor taught is used against its teacher.[16] It is essential that the decision maker forget what he or she has been taught at trial about the claimed invention and cast the mind back to the time the invention was made. . . to occupy the mind of one skilled in the art who is presented only with the references, and who is normally guided by the then-accepted wisdom in the art.[17]

---

11 *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

12 *In re Fritch*, 23 USPQ 2d 1780, 1784 (Fed. Cir. 1992) ("It is impermissible to use the claimed invention as an instruction manual or 'template' to piece together the teachings of the prior art so that the claimed invention is rendered obvious."); *Fromson v. Advance Offset Plate, Inc.*, 755 F.2d 1549, 1556, 225 USPQ 26, 31 (Fed. Cir. 1985) (nothing of record plainly indicated that it would have been obvious to combine previously separate lithography steps into one process). See e.g., *In re Gordon et al.*, 733 F.2d 900, 902, 221 USPQ 1125, 1127 (Fed. Cir. 1984) (mere fact that prior art could be modified by turning apparatus upside down does not make modification obvious unless prior art suggests desirability of modification); *Ex Parte Kaiser*, 194 USPQ 47, 48 (Pat. Bd. of App. 1975) (Examiner's failure to indicate anywhere in the record his reason for finding alteration of reference to be obvious militates against rejection).

13 *In re Keller*, 642 F.2d 413, 425, 208 USPQ 871, 881 (CCPA 1981).

14 *ACS Hosp. Sys. Inc. v. Montefiore Hosp.*, 32 F.2d 1572, 1577, 221 USPQ 929, 933 (Fed. Cir. 1984).

15 *Id.*

16 *W. L. Gore & Assoc. v. Garlock, Inc.*, 721 F.2d 1540, 1553, 220 USPQ 303, 312-313 (Fed. Cir. 1983), cert. denied, 469 U.S. 851 (1984).

17 *Id.*

The policy of the Patent and Trademark Office[18] is to follow in each and every case the standard of patentability enunciated by the Supreme Court in *Graham v. John Deere Co.*[19] As stated by the Supreme Court:

> Under ' 103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or non-obviousness of the subject matter is determined. Such secondary considerations as commercial success, long felt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented. As indicia of obviousness or nonobviousness, these inquiries may have relevancy.[20]

Thus, hindsight reconstruction, using the applicant's specification itself as a guide, is improper because it fails to consider the subject matter of the invention "as a whole" and fails to consider the invention as of the date at which the invention was made.

In addition to the reasons stated above, the *prima facie* obviousness case is further defective because the Office Action failed to show that there would be a reasonable expectation of success in modifying/combining references.[21] The Examiner bears the initial burden of factually supporting any *prima facie* conclusion of obviousness.[22] If the Examiner does not produce a prima facie case, the Applicant is under *no* obligation to submit evidence of nonobviousness.[23] Critically, to establish a *prima facie* case of

---

18 MPEP 2141.

19 148 USPQ 459 (1966).

20 148 USPQ at 467.

21 MPEP 2142, 2143.02.

22 MPEP 2142.

23 *Id.*

obviousness, *there must be a reasonable expectation of success.*[24] This reasonable expectation of success must be found in the prior art, not in Applicant's disclosure.[25]

The Office Action lacks any evidence, allegation, or mere mention of the legally required "reasonable expectation of success." Because this mandatory topic is unaddressed by the Office Action, no *prima facie* case of obviousness has been properly established.

As shown above, Claim 1 is patentable because a *prima facie* case of obviousness does not exist. Namely, (1) the applied art fails to teach the features of the claims, (2) there is insufficient motivation to combine/modify references as proposed by the Office Action, and (3) there is no showing that an ordinarily skilled artisan would have a reasonable expectation of success in making the office action's proposed modification of references.

Claim 8

Applicant restates that above argument for Claim 1 with regard to Claim 8, the claim limitations being similar and the bases for rejection also being similar. As with Claim 1, based upon the above reasoning, Claim 8 is patentably distinct from the Examiner's proposed combination of references, and reversal is appropriate.

Claim 15

Applicant restates that above argument for Claim 1 with regard to Claim 15, the claim limitations being similar and the bases for rejection also being similar. As with Claim 1, based upon the above reasoning, Claim 15 is patentably distinct from the Examiner's proposed combination of references, and reversal is appropriate.

Claims 2-4, 6, 7, 9-11, 13, 14, 16-18, 20, and 21

---

24 MPEP 2143.

25 *In re Vaeck*, 947 F.2d 488, 20 USPQ.2d 1438 (Fed. Cir. 1991). MPEP 2143.

Claims 2-4, 6, 7, 9-11, 13, 14, 16-18, 20, and 21 depend directly or indirectly from allowable independent Claims 1, 8, and 15, respectively. As a result, Claims 2-4, 6, 7, 9-11, 13, 14, 16-18, 20, and 21 should also be allowed at least for the same reasons as stated above with respect to Claims 1, 8, and 15.

Subparagraph (v) -- OTHER GROUNDS OF REJECTION

There are no other grounds of rejection.

CONCLUSION

For the foregoing reasons, the claims in the application are clearly and patentably distinguished over the cited references. Accordingly, the Examiner should be reversed and ordered to pass the case to issue.

If any fees are required by this submission, an appropriate fee submittal sheet is enclosed herewith. If fees are required yet this sheet is inadvertently missing, or the fees are incorrect in amount, please charge the charge the required fees (or credit any overpayment) to Deposit Account No. 07-1445.

Respectfully submitted,

Michael A. Glenn
Reg. No. 30,176

USPTO Customer 22,862

APPENDIX A

Applicant's Claims 1-4, 6-11, 13-18, 20, and 21

APPENDIX A

Applicant's Claims 1-4, 6-11, 13-18, 20, and 21

1. (previously amended) A process for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment, comprising the steps of:

providing a rule engine;

wherein said rule engine evaluates said business object;

providing administration means for allowing a user to maintain preferences for a specific instance of a business object;

wherein said rule engine uses business rules to evaluate a relationship between said business objects;

wherein each business object is a voter that provides votes that are evaluated by said business rules;

providing rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set; and

wherein a sequence of voters and an order of the votes included for each voter determine values in said solution set.

2. (original) The process of claim 1, further comprising the steps of:

providing domain tables;

wherein said tables are used to define the overall set of possible values for a given attribute of a business object; and

wherein said tables are static, configured, or dynamic in nature.

3. (original) The process of claim 1, wherein said business rules are configurable, generalized statements of how common processing methods are applied to a specific intersection of data; and wherein said business rules determine privileges, application of business processes, business relationships, choices, and default values.

4. (original) The process of claim 1, further comprising the step of:

providing rule administration means for allowing a user to define business rule attributes.

5. (canceled)

6. (original) The process of claim 1, wherein said rule engine returns one solution set for the set of business objects (voters) being referenced.

7. (original) The process of claim 1, wherein said business rules are cached.

8. (previously amended) An apparatus for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment, comprising:

a rule engine;

wherein said rule engine evaluates said business objects;

administration means for allowing a user to maintain preferences for a specific instance of a business object;

wherein said rule engine uses business rules to evaluate a relationship between said business objects;

wherein each business object is a voter that provides votes that are evaluated by said business rules;

rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set; and

wherein the sequence of voters and the order of the votes included for each voter determines values in said solution set.

9. (original) The apparatus of claim 8, further comprising:

domain tables;

wherein said tables are used to define the overall set of possible values for a given attribute of a business object; and

wherein said tables are static, configured, or dynamic in nature.

10. (original) The apparatus of claim 8, wherein said business rules are configurable, generalized statements of how common processing methods are applied to a specific intersection of data; and wherein said business rules determine privileges, application of business processes, business relationships, choices, and default values.

11. (original) The apparatus of claim 8, further comprising:

rule administration means for allowing a user to define business rule attributes.

12. (canceled)

13. (original) The apparatus of claim 8, wherein said rule engine returns one solution set for the set of business objects (voters) being referenced.

14. (original) The apparatus of claim 8, wherein said business rules are cached.

15. (previously amended) A program storage medium readable by a computer, tangibly embodying a program of instructions executable by the computer to perform method steps for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment, comprising the steps of:

providing a rule engine;

wherein said rule engine evaluates said business objects;

providing administration means for allowing a user to maintain preferences for a specific instance of a business object;

wherein said rule engine uses business rules to evaluate a relationship between said business objects;

wherein each business object is a voter that provides votes that are evaluated by said business rules;

providing rule resolution strategy means for resolving conflicts between specific

preferences of voters and determining a correct solution set; and

wherein the sequence of voters and the order of the votes included for each voter determines values in said solution set.

16. (original) The method of claim 15, further comprising the steps of:

providing domain tables;

wherein said tables are used to define the overall set of possible values for a given attribute of a business object; and

wherein said tables are static, configured, or dynamic in nature.

17. (original) The method of claim 15, wherein said business rules are configurable, generalized statements of how common processing methods are applied to a specific intersection of data; and wherein said business rules determine privileges, application of business processes, business relationships, choices, and default values.

18. (original) The method of claim 15, further comprising the step of:

providing rule administration means for allowing a user to define business rule attributes.

19. (canceled)

20. (original) The method of claim 15, wherein said rule engine returns one solution set for the set of business objects (voters) being referenced.

21. (original) The method of claim 15, wherein said business rules are cached.

APPENDIX B

Applicant's Figures 1-11

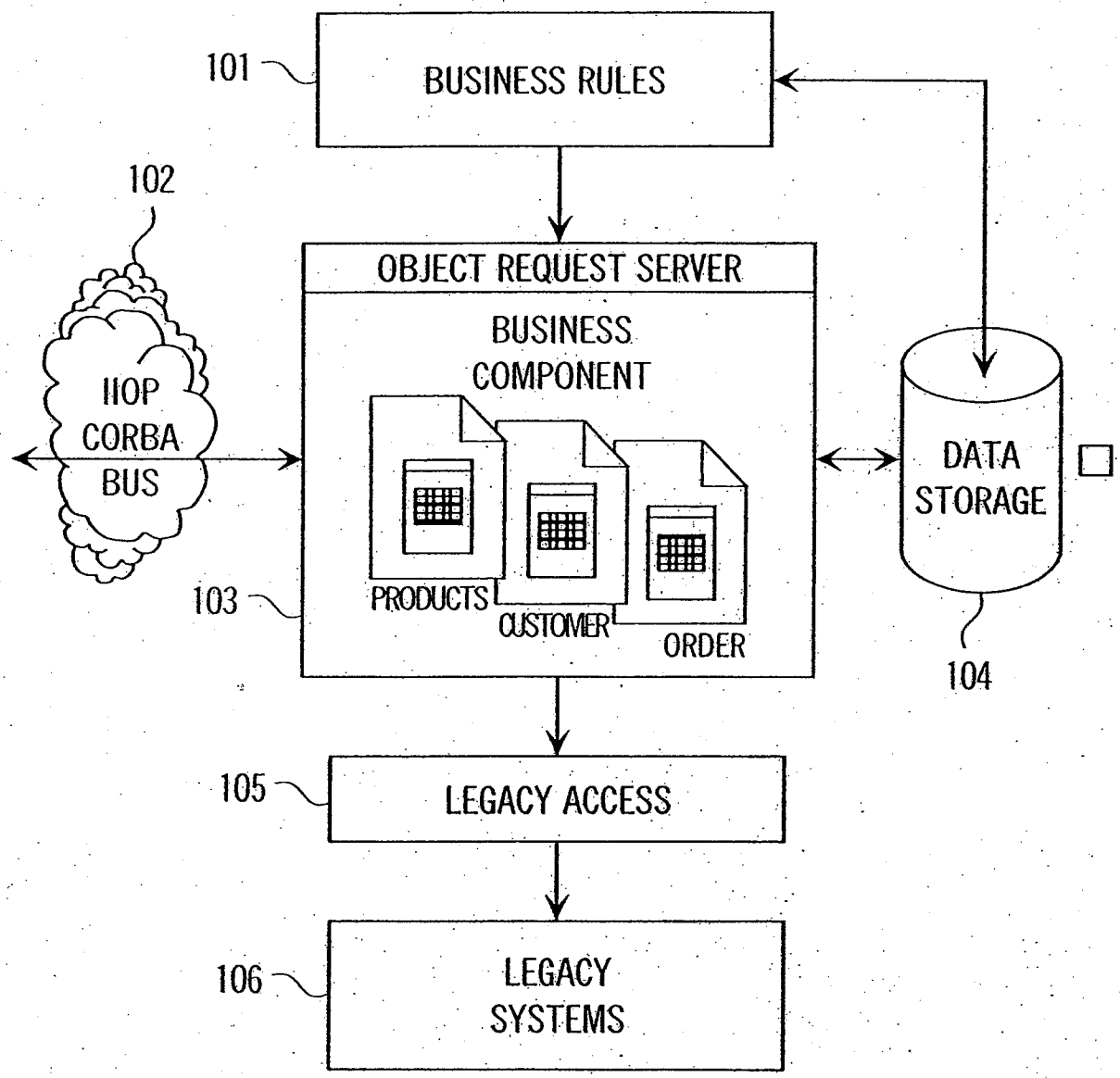

101
BUSINESS RULES
102
OBJECT REQUEST SERVER
BUSINESS
COMPONENT
IIOP
CORBA
BUS
PRODUCTS
CUSTOMER
ORDER
103
DATA
STORAGE
104
105
LEGACY ACCESS
106
LEGACY
SYSTEMS

FIG. 1

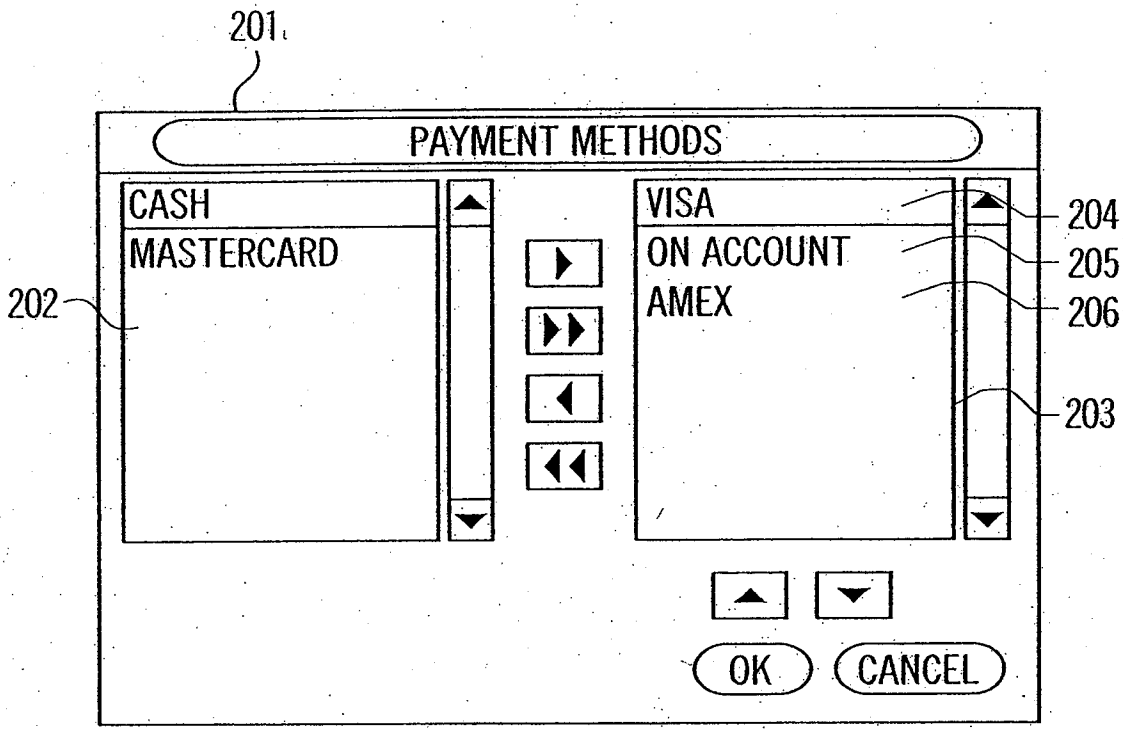


FIG. 2

301

VOTER	VOTES
SELLER COMPANY	VISA, MASTERCARD, AMERICAN EXPRESS, ON-ACCOUNT
BUYER COMPANY	VISA, AMERICAN EXPRESS, DISCOVER, CASH, ON-ACCOUNT
BUYER COMPANY PERSON	ON-ACCOUNT, VISA

302 304 306 303 305 307

FIG. 3

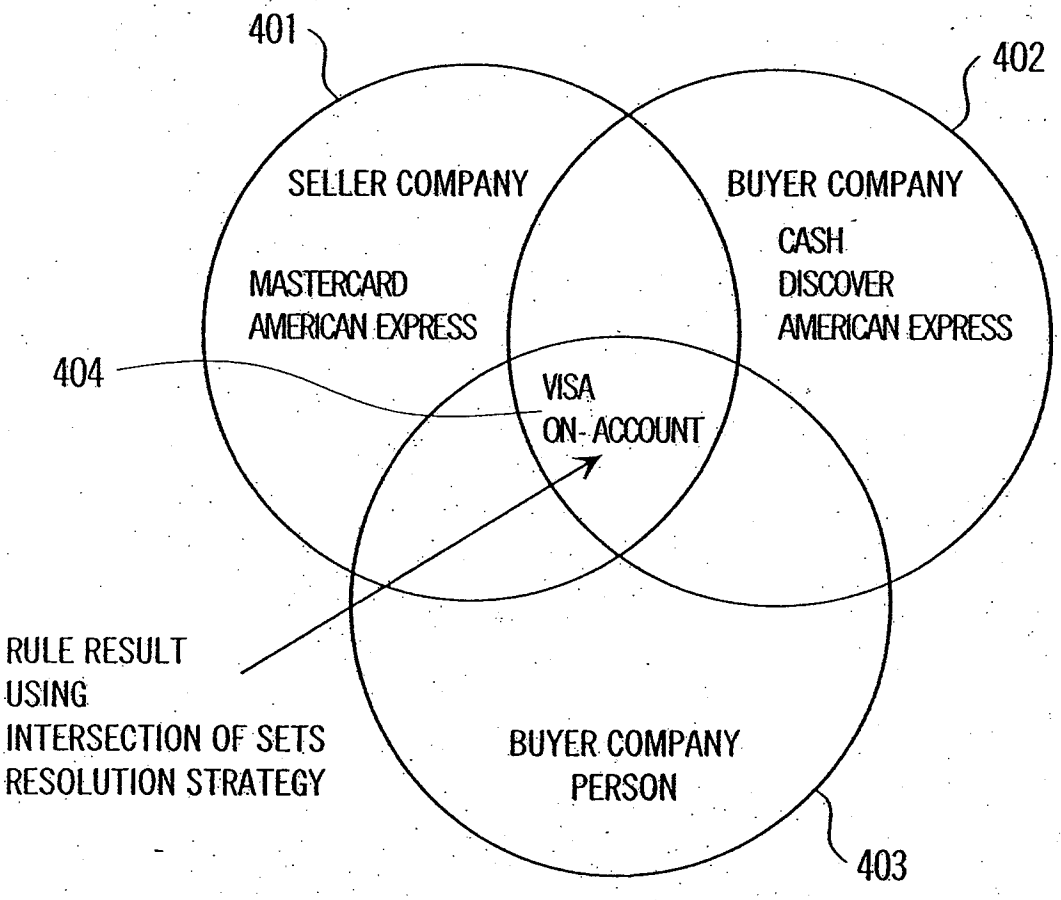

401
402
SELLER COMPANY
MASTERCARD
AMERICAN EXPRESS
BUYER COMPANY
CASH
DISCOVER
AMERICAN EXPRESS
404
VISA
ON-ACCOUNT
RULE RESULT
USING
INTERSECTION OF SETS
RESOLUTION STRATEGY
BUYER COMPANY
PERSON
403

FIG. 4

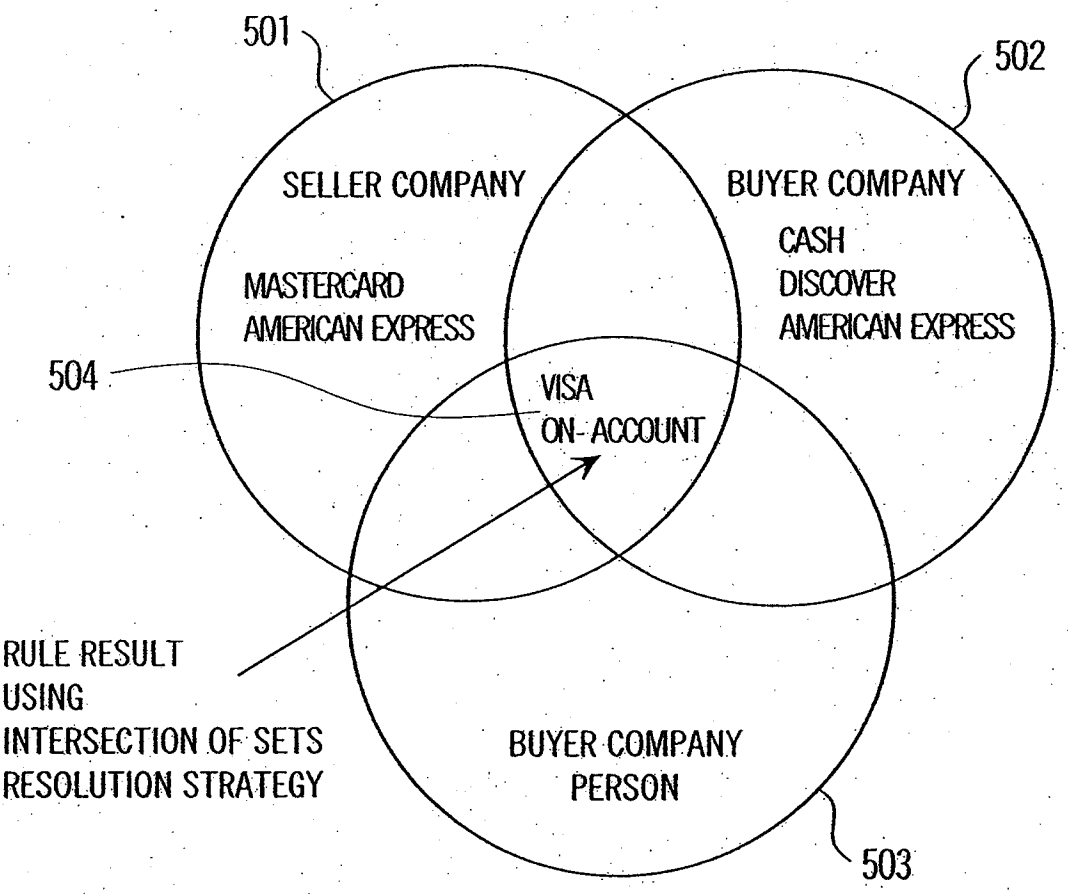

501
502
SELLER COMPANY
MASTERCARD
AMERICAN EXPRESS
BUYER COMPANY
CASH
DISCOVER
AMERICAN EXPRESS
504
VISA
ON-ACCOUNT
RULE RESULT
USING
INTERSECTION OF SETS
RESOLUTION STRATEGY
BUYER COMPANY
PERSON
503

FIG. 5A

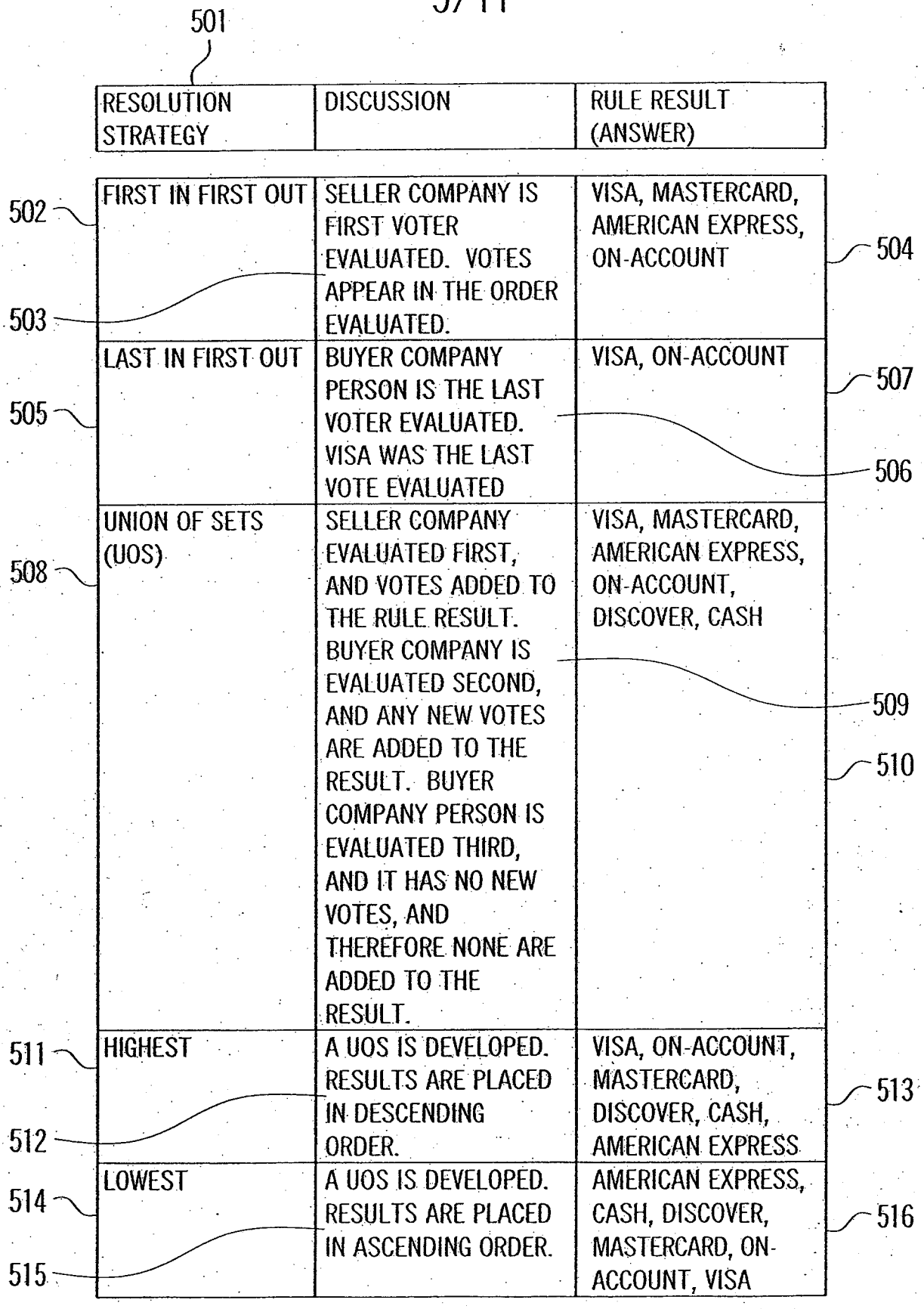

501
RESOLUTION STRATEGY
DISCUSSION
RULE RESULT (ANSWER)
502
FIRST IN FIRST OUT
SELLER COMPANY IS FIRST VOTER EVALUATED. VOTES APPEAR IN THE ORDER EVALUATED.
503
VISA, MASTERCARD, AMERICAN EXPRESS, ON-ACCOUNT
504
505
LAST IN FIRST OUT
BUYER COMPANY PERSON IS THE LAST VOTER EVALUATED. VISA WAS THE LAST VOTE EVALUATED
506
VISA, ON-ACCOUNT
507
508
UNION OF SETS (UOS)
SELLER COMPANY EVALUATED FIRST, AND VOTES ADDED TO THE RULE RESULT. BUYER COMPANY IS EVALUATED SECOND, AND ANY NEW VOTES ARE ADDED TO THE RESULT. BUYER COMPANY PERSON IS EVALUATED THIRD, AND IT HAS NO NEW VOTES, AND THEREFORE NONE ARE ADDED TO THE RESULT.
509
VISA, MASTERCARD, AMERICAN EXPRESS, ON-ACCOUNT, DISCOVER, CASH
510
511
HIGHEST
A UOS IS DEVELOPED. RESULTS ARE PLACED IN DESCENDING ORDER.
512
VISA, ON-ACCOUNT, MASTERCARD, DISCOVER, CASH, AMERICAN EXPRESS
513
514
LOWEST
A UOS IS DEVELOPED. RESULTS ARE PLACED IN ASCENDING ORDER.
515
AMERICAN EXPRESS, CASH, DISCOVER, MASTERCARD, ON-ACCOUNT, VISA
516

FIG. 5B

PAYMENT METHOD:	COMPANY DEFAULT	NEW...
CUSTOMER P.O. #:		
SHIP VIA:	COMPANY DEFAULT	
PREFERRED DELIVERY DATE (MM/DD/YY):	02/17/9	

TOTAL GOODS AND SERVICES:	287.60
ADJUSTMENTS:	0.00
NET TOTAL:	287.60
SHIPPING:	6.70
TAXABLE AMOUNT:	294.30
TAX:	22.07
TOTAL:	**316.37**

RECALCULATE

FIG. 6

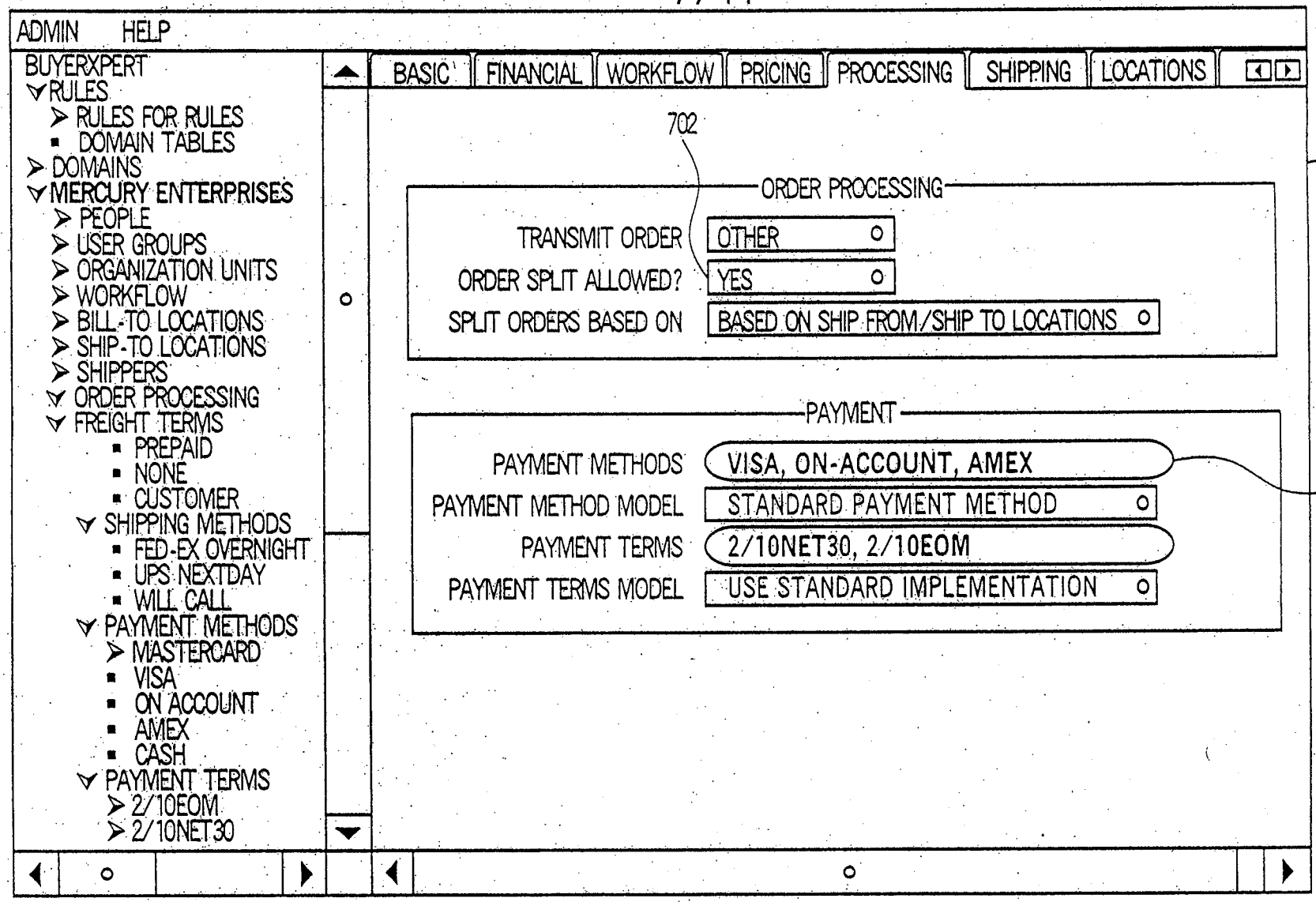

ADMIN
HELP
BUYERXPERT
RULES
RULES FOR RULES
DOMAIN TABLES
DOMAINS
MERCURY ENTERPRISES
PEOPLE
USER GROUPS
ORGANIZATION UNITS
WORKFLOW
BILL-TO LOCATIONS
SHIP-TO LOCATIONS
SHIPPERS
ORDER PROCESSING
FREIGHT TERMS
PREPAID
NONE
CUSTOMER
SHIPPING METHODS
FED-EX OVERNIGHT
UPS NEXTDAY
WILL CALL
PAYMENT METHODS
MASTERCARD
VISA
ON ACCOUNT
AMEX
CASH
PAYMENT TERMS
2/10EOM
2/10NET30
BASIC
FINANCIAL
WORKFLOW
PRICING
PROCESSING
SHIPPING
LOCATIONS
702
701
ORDER PROCESSING
TRANSMIT ORDER
OTHER
ORDER SPLIT ALLOWED?
YES
SPLIT ORDERS BASED ON
BASED ON SHIP FROM/SHIP TO LOCATIONS
PAYMENT
PAYMENT METHODS
VISA, ON-ACCOUNT, AMEX
703
PAYMENT METHOD MODEL
STANDARD PAYMENT METHOD
PAYMENT TERMS
2/10NET30, 2/10EOM
PAYMENT TERMS MODEL
USE STANDARD IMPLEMENTATION

FIG. 7

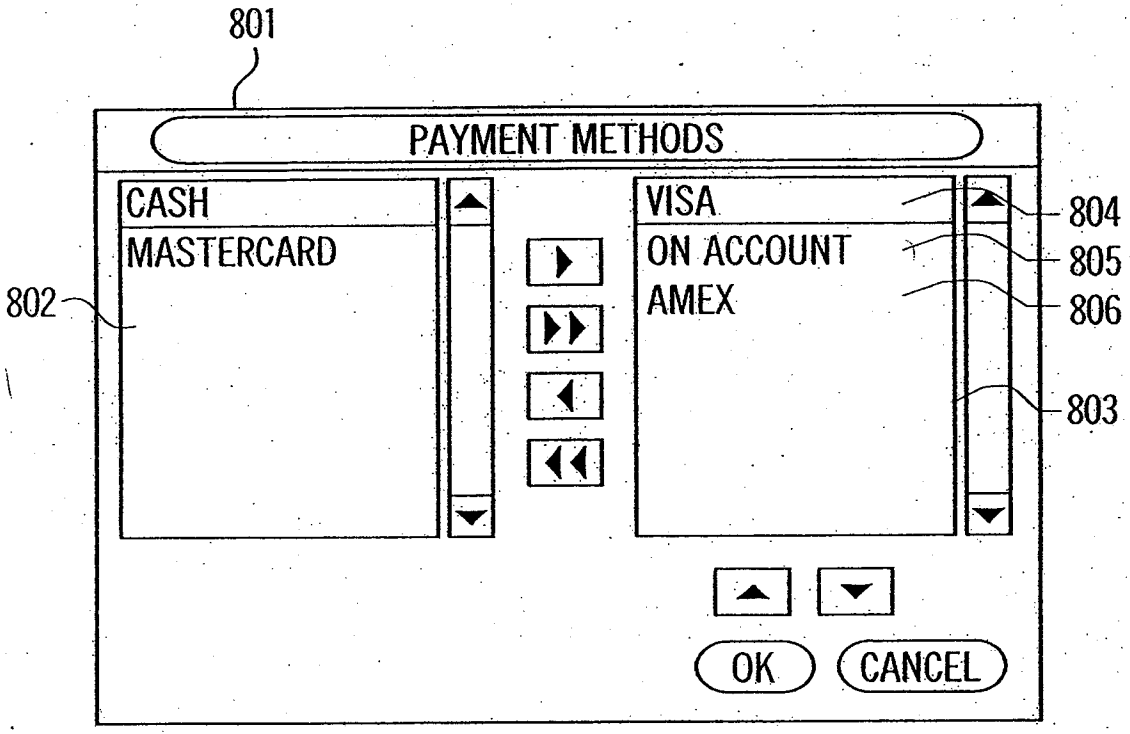

801
PAYMENT METHODS
CASH
MASTERCARD
802
VISA
ON ACCOUNT
AMEX
804
805
806
803
OK
CANCEL

FIG. 8

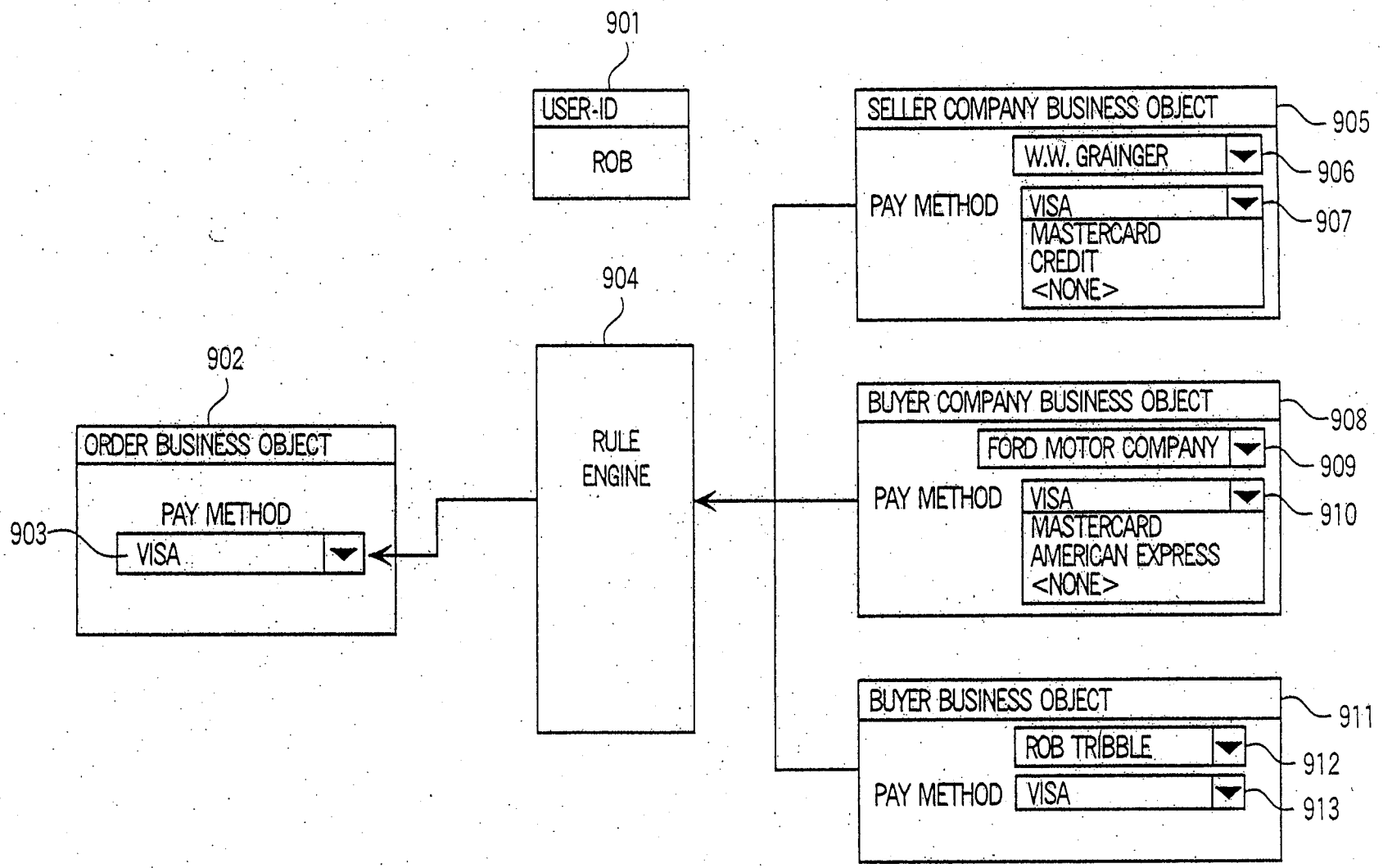

901
USER-ID
ROB
SELLER COMPANY BUSINESS OBJECT
905
W.W. GRAINGER
906
PAY METHOD
VISA
907
MASTERCARD
CREDIT
<NONE>
904
902
ORDER BUSINESS OBJECT
RULE ENGINE
PAY METHOD
903
VISA
BUYER COMPANY BUSINESS OBJECT
908
FORD MOTOR COMPANY
909
PAY METHOD
VISA
910
MASTERCARD
AMERICAN EXPRESS
<NONE>
BUYER BUSINESS OBJECT
911
ROB TRIBBLE
912
PAY METHOD
VISA
913

FIG. 9

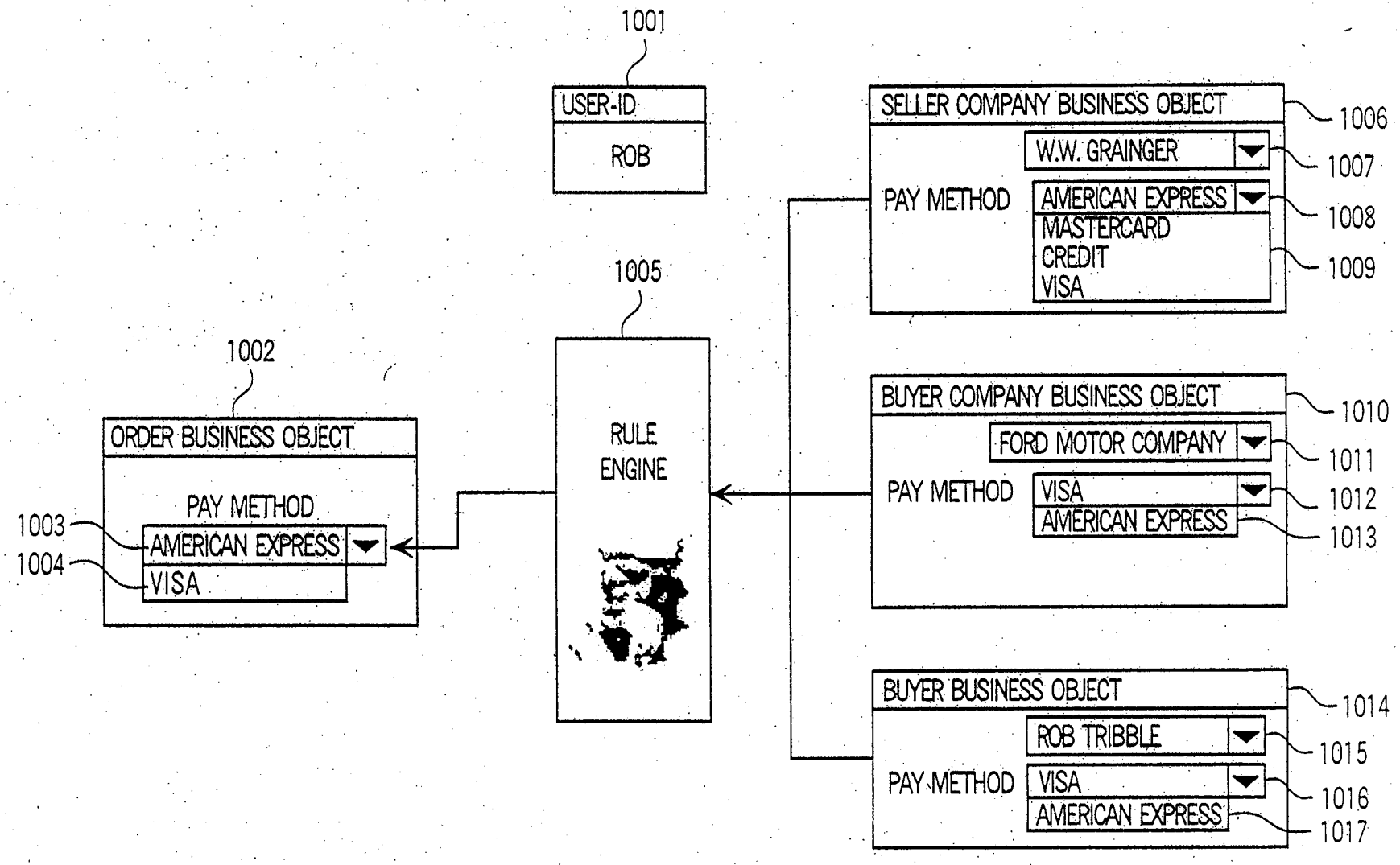

1001
USER-ID
ROB
SELLER COMPANY BUSINESS OBJECT
1006
W.W. GRAINGER
1007
PAY METHOD
AMERICAN EXPRESS
1008
MASTERCARD
CREDIT
VISA
1009
1005
1002
ORDER BUSINESS OBJECT
RULE ENGINE
BUYER COMPANY BUSINESS OBJECT
1010
FORD MOTOR COMPANY
1011
PAY METHOD
VISA
1012
AMERICAN EXPRESS
1013
PAY METHOD
1003
AMERICAN EXPRESS
1004
VISA
BUYER BUSINESS OBJECT
1014
ROB TRIBBLE
1015
PAY METHOD
VISA
1016
AMERICAN EXPRESS
1017

FIG. 10

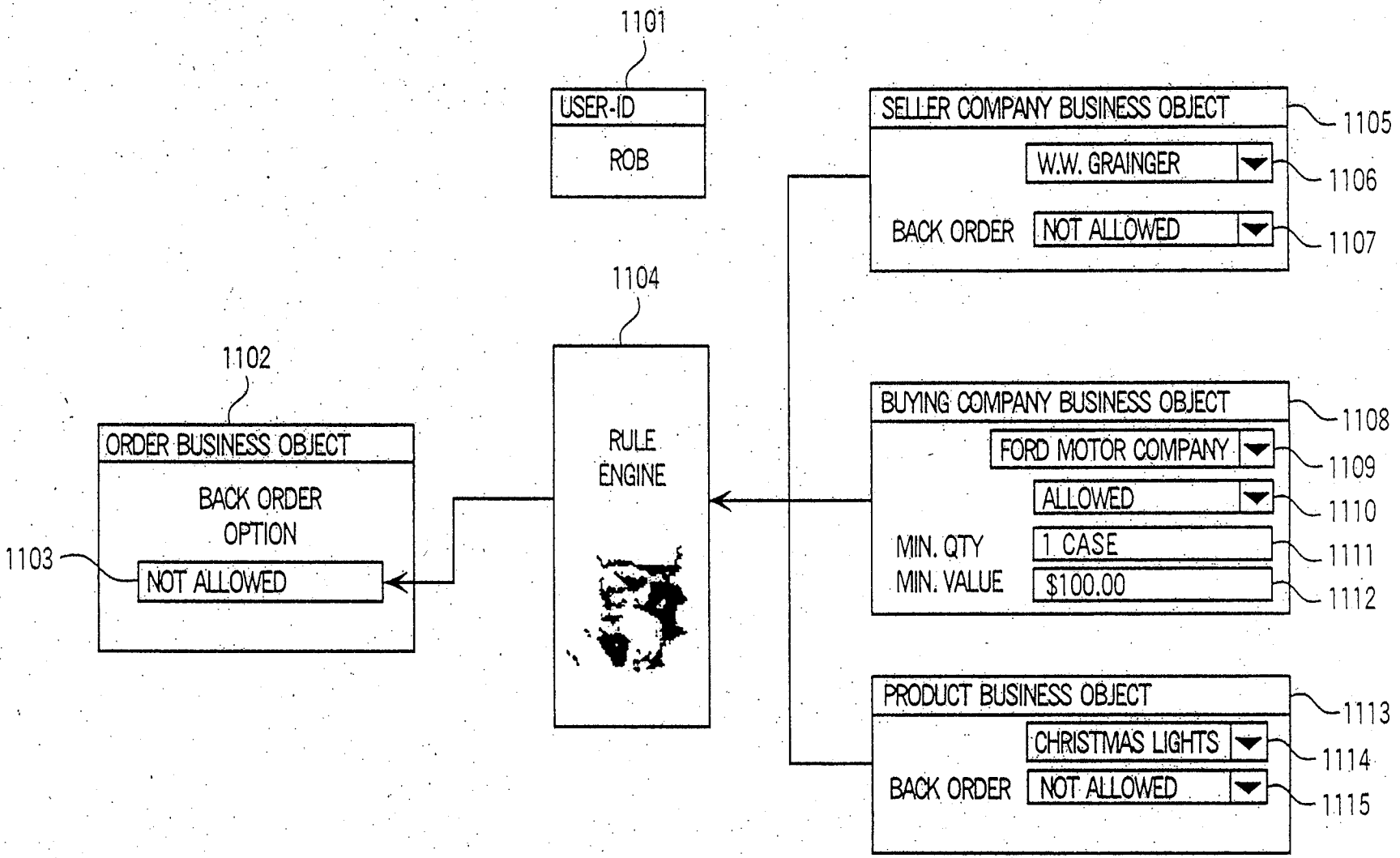

1101
USER-ID
ROB
SELLER COMPANY BUSINESS OBJECT
1105
W.W. GRAINGER
1106
BACK ORDER
NOT ALLOWED
1107
1104
1102
ORDER BUSINESS OBJECT
BACK ORDER
OPTION
1103
NOT ALLOWED
RULE
ENGINE
BUYING COMPANY BUSINESS OBJECT
1108
FORD MOTOR COMPANY
1109
ALLOWED
1110
MIN. QTY
1 CASE
1111
MIN. VALUE
$100.00
1112
PRODUCT BUSINESS OBJECT
1113
CHRISTMAS LIGHTS
1114
BACK ORDER
NOT ALLOWED
1115

FIG. 11

APPENDIX C

Evidences

(none)

APPENDIX D

Related Appeals and Interferences

(none)

APPENDIX E

Copy of the Non-Final Office Action dated 8/27/04

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/668,125	09/21/2000	Rob Tribble	NETS0044	1382

22862 7590 08/27/2004

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

EXAMINER	
FRENEL, VANEL	

ART UNIT	PAPER NUMBER
3626	

DATE MAILED: 08/27/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

GPG
U.S.: ✓ FOREIGN:
DOCKETED: 9/2/04 BY: EL
ACTION: Response Due
DUE DATE: 11/27/04
EXT: 1ST 12/27 2ND 1/27 3RD 2/27/05
DOCKET# NETS0044 ATTY: AC

ACKNOWLEDGE RECEIPT
SEP - 1 2004
GLENN PATENT GROUP

Office Action Summary	Application No.	Applicant(s)	
	09/668,125	TRIBBLE, ROB	
	Examiner	Art Unit	
	Vanel Frenel	3626	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *7/7/04*.

2a)☐ This action is FINAL. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,6-11,13-18,20 and 21* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4,6-11,13-18,20 and 21* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☑ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Notice to Applicant

1. This communication is in response to the Amendment filed 2/10/04. Claims 1-4, 6-11, 13-18 and 20-21 are pending.

2. Applicant's request for reconsideration of the finality of the rejection of the last Office action is persuasive and, therefore, the finality of that action is withdrawn.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-4, 6-11, 13-18 and 20-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chisholm (5,400,248) in view of Marsh et al (4,210,962).

(A) As per claim 1, Chisholm discloses the steps of:

providing a rule engine (See Chisholm, Col.5, lines 17-34 The Examiner interprets Chisholm's voter administrator program to be a form of "rules engine");

wherein said rule engine evaluates said business objects (Col.5, lines 23-34 The Examiner interprets "votes" as "business objects");

providing administration means for allowing a user to maintain preferences for a specific instance of a business objective (Col.5, lines 12-34);

wherein said rule engine uses business rules to evaluate a relationship between said business objects (Col.5, lines 55-68 The Examiner understands conditional votes are votes having a relationship to other votes); and

wherein each business object is a voter that provides votes that are evaluated by said business rules (Col.5, lines 12-34);

providing rule resolution strategy means for resolving conflicts between specific preferences of voters and determining a correct solution set (Col.5, lines 55-65; Col.8, lines 10-44);

wherein a sequence of voters and an order of the votes included for each voter determines values in said solution set (Col.9, lines 35-50; Col.10, lines 55-68 to Col.11, line 3).

Chisholm does not explicitly disclose a process for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment, as recited in the preamble of claim 1.

However, this feature is known in the art, as evidenced by Marsh. In particular, Marsh suggests a process for evaluating business objects with no prior association and creating dynamic solution sets based on said evaluation in a computer environment (See Marsh, Col.3, lines 8-11; Col.4, lines 22-28).

It would have been obvious to one of ordinary skill in the art at the time of the invention to have included the feature of Marsh within the system of Chisholm with the motivation of solving dynamic programs in a minimum of time and without the necessity of complex interprocessor communication (See Marsh Col.2, lines 21-25).

(B) As per claim 2, Marsh discloses the process further comprising the steps of: providing domain tables (Col.4, lines 28-55); wherein said tables are used to define the overall set of possible values for a given attribute of a business object (Col.5, lines 66 to Col.6, line 31 and Figs 1a-1c); and wherein said tables are static, configured, or dynamic in nature (Col.4, lines 22-28; Col.5, lines 66 to Col.6, line 31 and Figs 1a-1c).

(C) As per claim 3, Chisholm discloses the process wherein said business rules are configurable, generalized statements of how common processing methods are applied to a specific intersection of data (Col.10, lines 4-30); and wherein said business rules determine privileges, application of business processes, business relationships, choices, and default values (Col.12, lines 1-45).

(D) As per claim 4, Chisholm discloses the process further comprising the step of: providing rule administration means for allowing a user to define business rule attributes (Col.16, lines 61-68 to Col.17, line 7).

(E) As per claim 6, Chisholm discloses the process wherein said rule engine returns one solution set for the set of business objects (voters) being referenced (Col.8, lines 30-68).

(F) As per claim 7, Chisholm discloses the process wherein said business rules are cached (Col.5, lines 10-47).

(G) Apparatus Claims 8-14 differ from method claims 1-7 by reciting an apparatus for performing the underlying process steps of method claims 1-7. Since Chisholm clearly discloses the underlying process steps recited in claims 1-7, it is readily apparent that Chisholm discloses the necessary apparatus for performing those steps. Note, for example, the recitation in Chisholm with regard to a programming interface and a network (Chisholm, Col.4, line 65 to Col.5, line 10).

The remainder of claims 8-14 repeat the same limitations addressed above in the rejections of claims 1-7, and are therefore rejected for the same reasons given for claims 1-7.

(H) Claims 15-21 differ from claims 1-7 by reciting a "program storage medium readable, tangibly embodying a program of instructions". As per this limitation, Note, for example, the recitation in Chisholm with regard to a programming interface and a network (Chisholm, Col.4, line 65 to Col.5, line 10).

The remainder of claims 15-21 repeat the same limitations addressed above in the rejections of claims 1-7, and are therefore rejected for the same reasons given for claims 1-7.

Response to Arguments

5. Applicant's arguments with respect to claims 1-4, 6-11, 13-18 and 20-21 have been considered but are moot in view of the new ground(s) of rejection.

Applicant's arguments with regard to the teachings of Fisk and Chacker are moot, as these references are not applied against the pending claims Rather, it is the collective teachings of Chisholm and White that obviate the presently pending claims.

Conclusion

6. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. The cited but not applied art teaches electronic trusted party (5,117,358).

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Vanel Frenel whose telephone number is 703-305-4952. The examiner can normally be reached on Monday-Thursday from 6:30am-5:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Joseph Thomas can be reached on 703-305-9643. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

V.F
V.F

August 20, 2004

JOSEPH THOMAS
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 3600

Notice of References Cited	Application/Control No.: 09/668,125	Applicant(s)/Patent Under Reexamination TRIBBLE, ROB	
	Examiner Vanel Frenel	Art Unit 3626	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-5,400,248	03-1995	Chisholm, John D.	705/12
	B	US-2002/0091550	07-2002	White et al.	705/4
	C	US-5,117,358	05-1992	Winkler, Peter M.	708/135
	D	US-4,210,962	07-1980	Marsh et al.	705/7
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

APPENDIX F

Copy of the Final Office Action dated 2/10/05

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/668,125	09/21/2000	Rob Tribble	NETS0044	1382

22862 7590 02/10/2005

GLENN PATENT GROUP
3475 EDISON WAY, SUITE L
MENLO PARK, CA 94025

ACKNOWLEDGE RECEIPT

FEB 14 2005

GLENN PATENT GROUP

EXAMINER	
FRENEL, VANEL	
ART UNIT	PAPER NUMBER
3626	

DATE MAILED: 02/10/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

GPG
U.S.: ✓ FOREIGN: _____
DOCKETED: 2/15/05 BY: GL
ACTION: Final Rejection
DUE DATE: 2mo 4/10 3mo 5/10/05
EXT: 1ST 6/10 2ND 7/10 3RD 8/10/05
DOCKET# NETS0044 ATTY: AC

Office Action Summary	Application No.	Applicant(s)	
	09/668,125	TRIBBLE, ROB	
	Examiner	Art Unit	
	Vanel Frenel	3626	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 November 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,6-11,13-18,20 and 21* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4, 6-11, 13-18, 20-21* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Notice to Applicant

1. This communication is in response to the Amendment filed on 11/29/04. Claims 1-4, 6-11, 13-18 and 20-21 are pending.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-4, 6-11, 13-18 and 20-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chisholm (5,400,248) in view of Marsh et al (4,210,962), for substantially the same reasons given in the prior Office Action, and incorporated herein. Further reasons appear hereinbelow.

(A) Claims 1-4, 6-11, 13-18 and 20-21 have not been amended and are rejected for the same reasons given in the prior Office Action mailed 8/27/04, and incorporated herein.

Response to Arguments

4. Applicant's arguments filed on 11/29/04 with respect to claims 1-4, 6-11, 13-18 and 20-21 have been fully considered but they are not persuasive. Applicant's arguments will be addressed in the order they appear hereinbelow.

At pages 2-4 of the 11/29/04 response, Applicant's argues the followings:

(1) Chilsholm and Marsh, taken alone or in combination, fail to teach, suggest, or render obvious the present invention as claimed.

(2) Chilsholm and Marsh fail to teach or suggest providing a rule engine, which evaluates a relationship objects with no prior association and uses business rules to evaluate a relationship between the business objects, each business object being a voter that provides votes that are evaluated by the business rules, wherein a sequence of voters and an order of the votes determine values in a solution set, as claimed in independent claims 1,8, and 15.

(3) Impermissible hindsight to combine Chilshom and Marsh.

(A) With respect to Applicant's first argument, Examiner respectfully submits that obviousness is not determined on the basis of the evidence as a whole and the relative persuasiveness of the arguments. See In *re Oetiker*, 977F. 2d 1443, 1445,24 USPQ2d 1443, 1444 (Fed. Cir. 1992); In *re Hedges*, 783F.2d 1038, 1039, 228 USPQ 685, 686 (Fed. Cir.1992); In *re Piaseckii*, 745 F.2d 1468, 1472, 223 USPQ 785, 788 (Fed. Cir.1984); In *re Rinehart*, 531 F.2d 1048, 1052, 189 USPQ 143, 147 (CCPA 1976). Using this standard, the Examiner respectfully submits that he has at least satisfied the burden of presenting a prima facie case of obviousness, since he has presented evidence of corresponding claim elements in the prior art and has expressly articulated the combinations and the motivations for combinations that fairly suggest Applicant's claimed invention. Note, for example, in the instant case, the Examiner respectfully

notes that each and every motivation to combine the applied references are accompanied by select portions of the respective reference(s) which specially support that particular motivation and /or an explanation based on the logic and scientific reasoning of one ordinarily skilled in the art at the time of the invention that support a holding of obviousness. As such, it is not seen that the Examiner's combination of references is unsupported by the applied prior art of record. Rather, it is respectfully submitted that explanation based on the logic and scientific reasoning of one of ordinarily skilled in the art at the time of the invention that support a holding of obviousness has been adequately provided by the motivations and reasons indicated by the Examiner, Ex parte Levengood, 28 USPQ2d 1300(Bd. Pat. App.& Inter., 4/22/93). Therefore, the combination of references is proper and the rejection is maintained.

In addition, the Examiner recognizes that references cannot be arbitrarily altered or modified and that there must be some reason why one skilled in the art would be motivated to make the proposed modifications. However, although the Examiner agrees that the motivation or suggestion to make modifications must be articulated, it is respectfully contended that there is no requirement that the motivation to make modifications must be expressly articulated within the references themselves. References are evaluated by what they suggest to one versed in the art, rather than by their specific disclosures, In *re Bozek*, 163 USPQ 545 (CCPA 1969). Therefore, Applicant's argument is not persuasive.

(B) With respect to Applicant's second argument, Examiner respectfully submits that

He has relied upon Chisholm for the teaching of "a rule engine" which correspond to voter administrator program (See Chisholm, Col.5, lines 17-34). In addition, Examiner respectfully submits that He has relied upon Chisholm for the teaching of "each business object being a voter (See Chisholm, Col.5, lines 12-34) that provides votes that are evaluated by the business rules, wherein a sequence of voters and an order of the votes determine values in a solution set (See Chisholm, Col.9, lines 35-50; Col.10, lines 55-68 to Col.11, line 3) which correspond to Applicant claimed feature. Therefore, Applicant argument is not persuasive.

With regard to the teaching of "evaluating a relationship objects with no prior association and uses business rules". Examiner respectfully submits that Marsh suggests "dynamic programming is an approach for solving optimization (maximization or minimization) problems, relying on dissecting the main optimization problem into many intermediate optimization problems" since Marsh has been using variables which provide an absolute optimum transition in each state space (See Marsh, Col.3, lines 8-11; Col.4, lines 22-28) which correspond to Applicant's claimed feature. Therefore, Applicant argument is not persuasive.

(C) In response to Applicant's argument that the Examiner's conclusion of obviousness is based upon improper hindsight reasoning, it must be recognized that any judgment on obviousness is in a sense necessarily a reconstruction based upon hindsight reasoning. But so long as it takes into account only knowledge which was within the level of ordinary skill at the time the claimed invention was made, and does

not include knowledge gleaned only from the Applicant's disclosure, such a reconstruction is proper. See In re *McLaughlin,* 443 F.2d 1392, 170 USPQ 209 (CCPA 1971).

5. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Conclusion

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Vanel Frenel whose telephone number is 703-305-4952. The examiner can normally be reached on Monday-Thursday from 6:30am-5:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Joseph Thomas can be reached on 703-305-9588. The fax phone numbers

for the organization where this application or proceeding is assigned are 703-305-7687 for regular communications and 703-305-7687 for After Final communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is 703-308-1113.

V.F
V.F

January 31, 2005

JOSEPH THOMAS
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 3600

APPENDIX G

USPN 5,400,248

US005400248A

United States Patent [19]

Chisholm

[11] Patent Number: 5,400,248

[45] Date of Patent: Mar. 21, 1995

[54] COMPUTER NETWORK BASED CONDITIONAL VOTING SYSTEM

[75] Inventor: **John D. Chisholm**, 140 Sand Hill Circle, Menlo Park, Calif.

[73] Assignee: **John D. Chisholm**, Menlo Park, Calif.

[21] Appl. No.: **122,869**

[22] Filed: **Sep. 15, 1993**

[51] **Int. Cl.**[6] **G06F 15/20**; G06F 15/30

[52] **U.S. Cl.** **364/409**; 235/386

[58] **Field of Search** 364/401, 402, 407, 409; 235/386, 456, 51, 385

[56] **References Cited**

U.S. PATENT DOCUMENTS

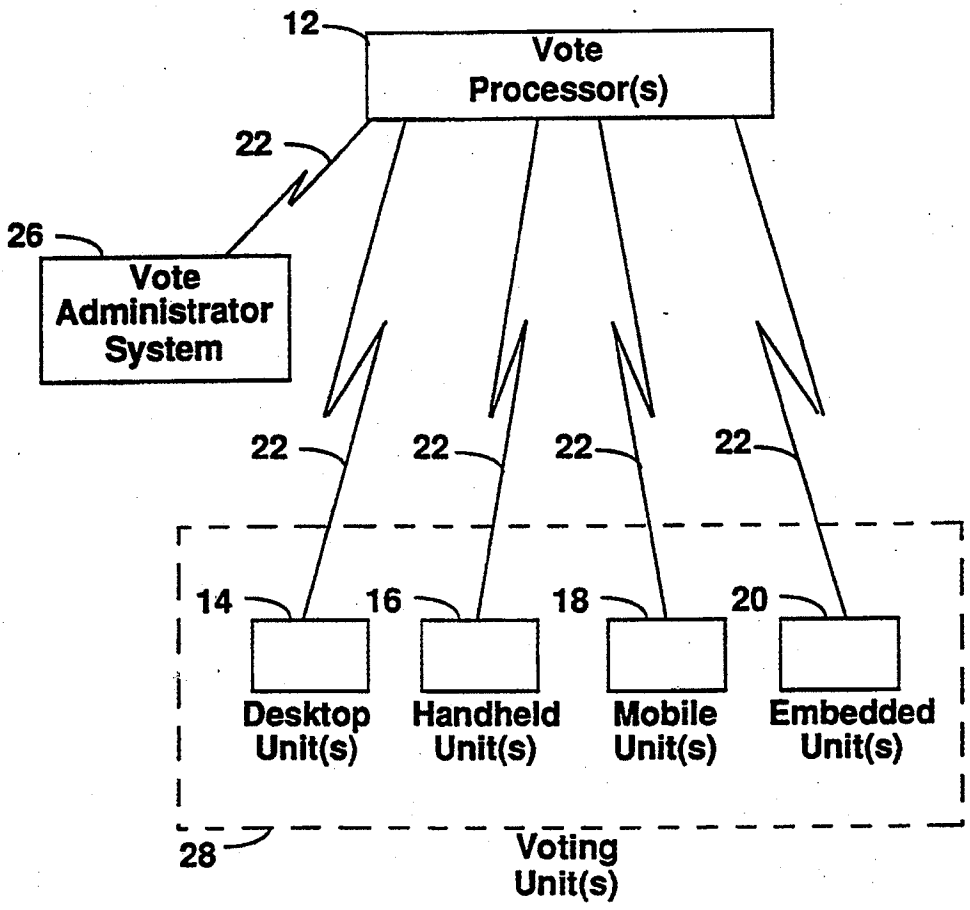
4,641,241	2/1987	Boram	364/409
4,774,665	9/1988	Webb	364/409
5,117,358	5/1992	Winkler	364/419
5,218,528	6/1993	Wise et al.	364/409
5,278,753	1/1994	Graft, III	364/409

Primary Examiner—Roy N. Envall, Jr.
Assistant Examiner—Khai Tran
Attorney, Agent, or Firm—Townsend and Townsend Khourie and Crew

[57] **ABSTRACT**

A voting system allows voters to express and cast votes that are conditional on the votes of others of a voting group. Votes may be conditional on the votes of specific individuals, on the number or percent of the overall group who vote a certain way, external events or on any combination thereof. The system solves the "common goods, free rider" dilemma in which voters oppose proposals they recognize as worthwhile out of fear that a few supporters will be burdened with all of the costs. The system specifies and enforces terms under which conditional voting will take place, and may manage the voting process across a network. The system recognizes when either multiple solutions or no solutions to a set of votes exist. The system can determine which voters are responsible for these cases, and can invite them to change their votes, if they wish. The system can also determine the largest subset or subsets of a group of conditional votes that has no solution, for which there is a unique solution or multiple solutions. Overall, the system leads to better and faster group decisions that are based on more complete voter knowledge than simply yes, no or abstain.

39 Claims, 13 Drawing Sheets



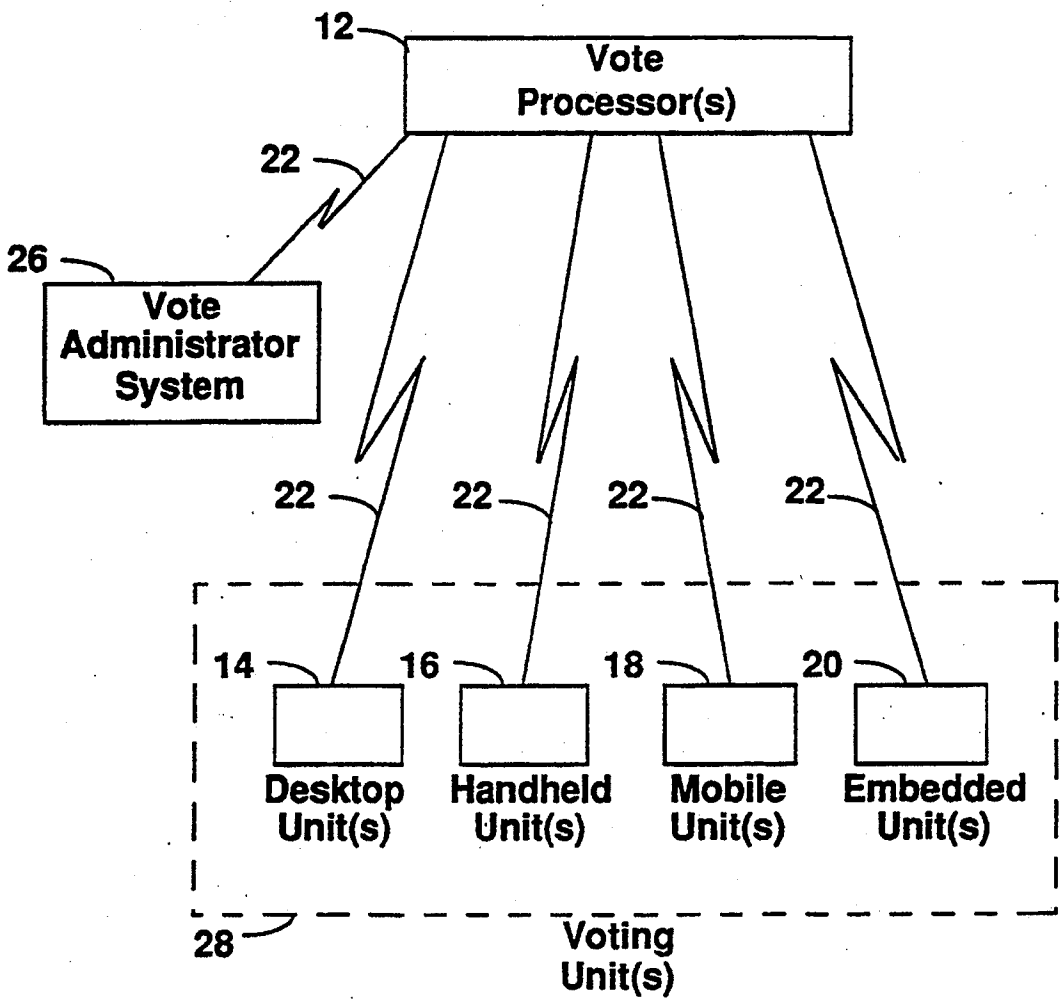

12
Vote Processor(s)
22
26
Vote Administrator System
22
22
22
22
14
16
18
20
Desktop Unit(s)
Handheld Unit(s)
Mobile Unit(s)
Embedded Unit(s)
28
Voting Unit(s)

Fig. 1

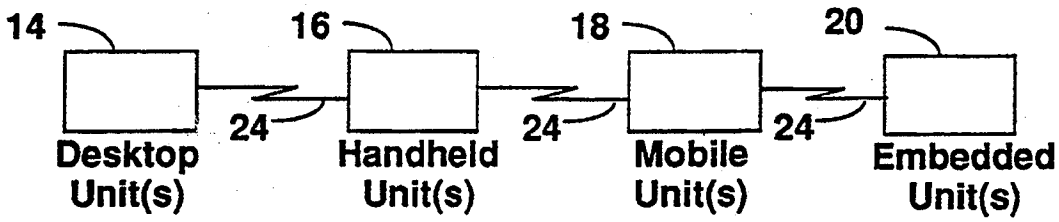

14
16
18
20
24
24
24
Desktop Unit(s)
Handheld Unit(s)
Mobile Unit(s)
Embedded Unit(s)

Fig. 2

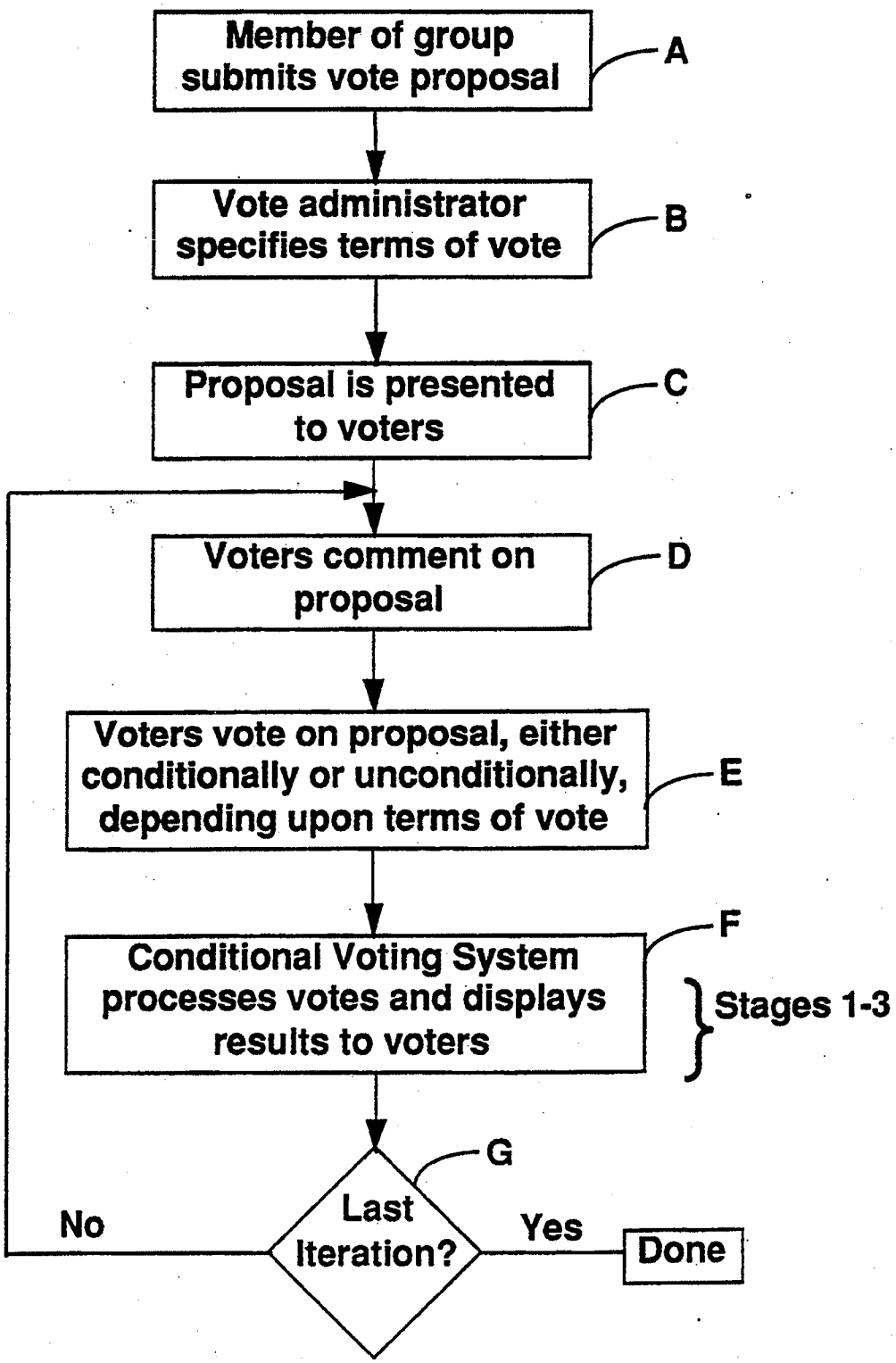

Member of group submits vote proposal
A
Vote administrator specifies terms of vote
B
Proposal is presented to voters
C
Voters comment on proposal
D
Voters vote on proposal, either conditionally or unconditionally, depending upon terms of vote
E
Conditional Voting System processes votes and displays results to voters
F
Stages 1-3
G
Last Iteration?
No
Yes
Done

Fig. 3

1) UNCONDITIONAL	CONDITIONAL	
No Dependencies • Yes • No • Abstain • No Vote	**2) *Group* Dependencies Only** (Voters are indistinguishable from one another)	**3) *Individual Voter* Dependencies** (Votes may depend upon votes of specific individuals)

Fig. 4

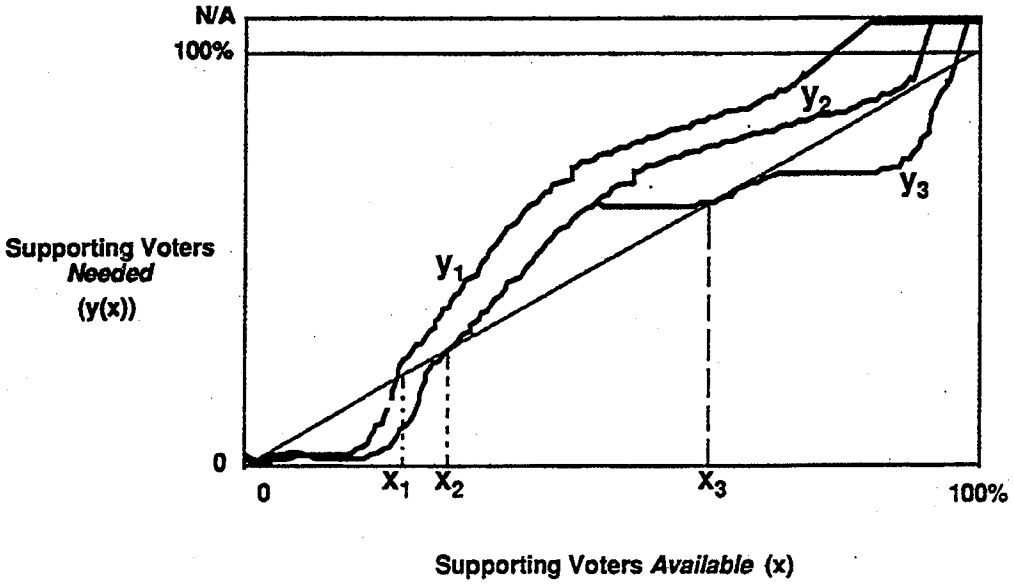


Fig. 5

Person:	Amy	Bill	Charlie	Dave	Ed	Frank
y = number of supporters in group required for this person to support proposal:	5	1	1	2	6	Not Applicable (NA)

Fig. 6

Person:	Bill	Charlie	Dave	Amy	Ed	Frank
x:	1	2	3	4	5	6
y = number of supporters in group required for this person to support proposal:	1	1	2	5	6	Not Applicable (NA)

Fig. 7

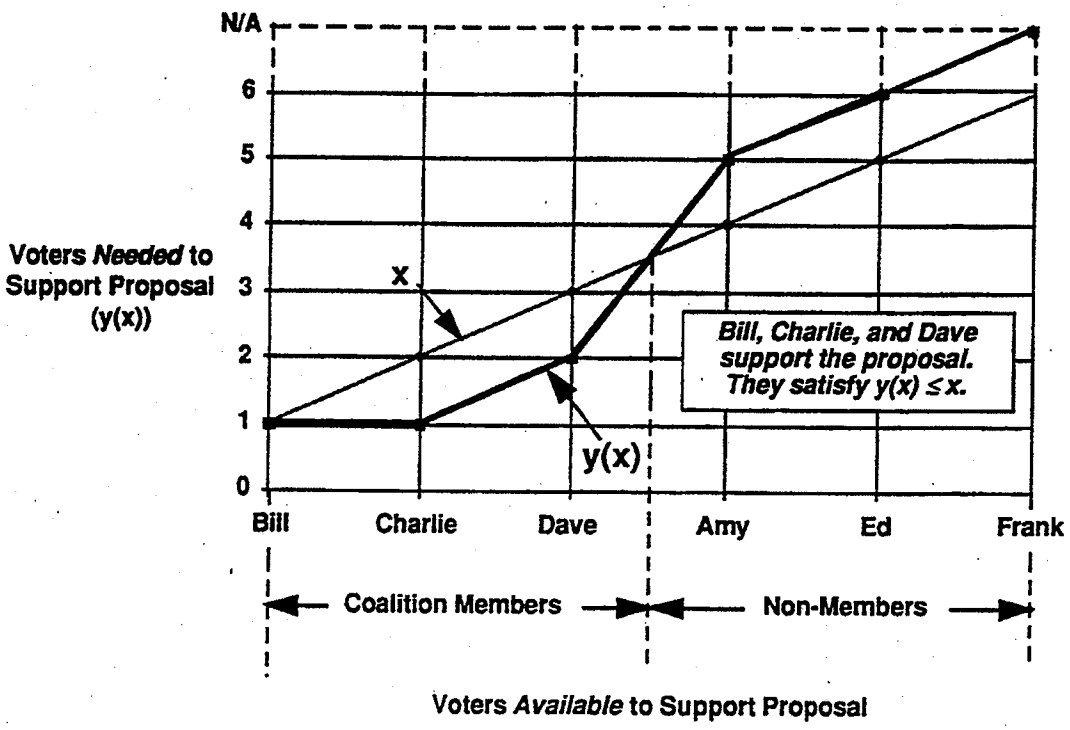


Fig. 8

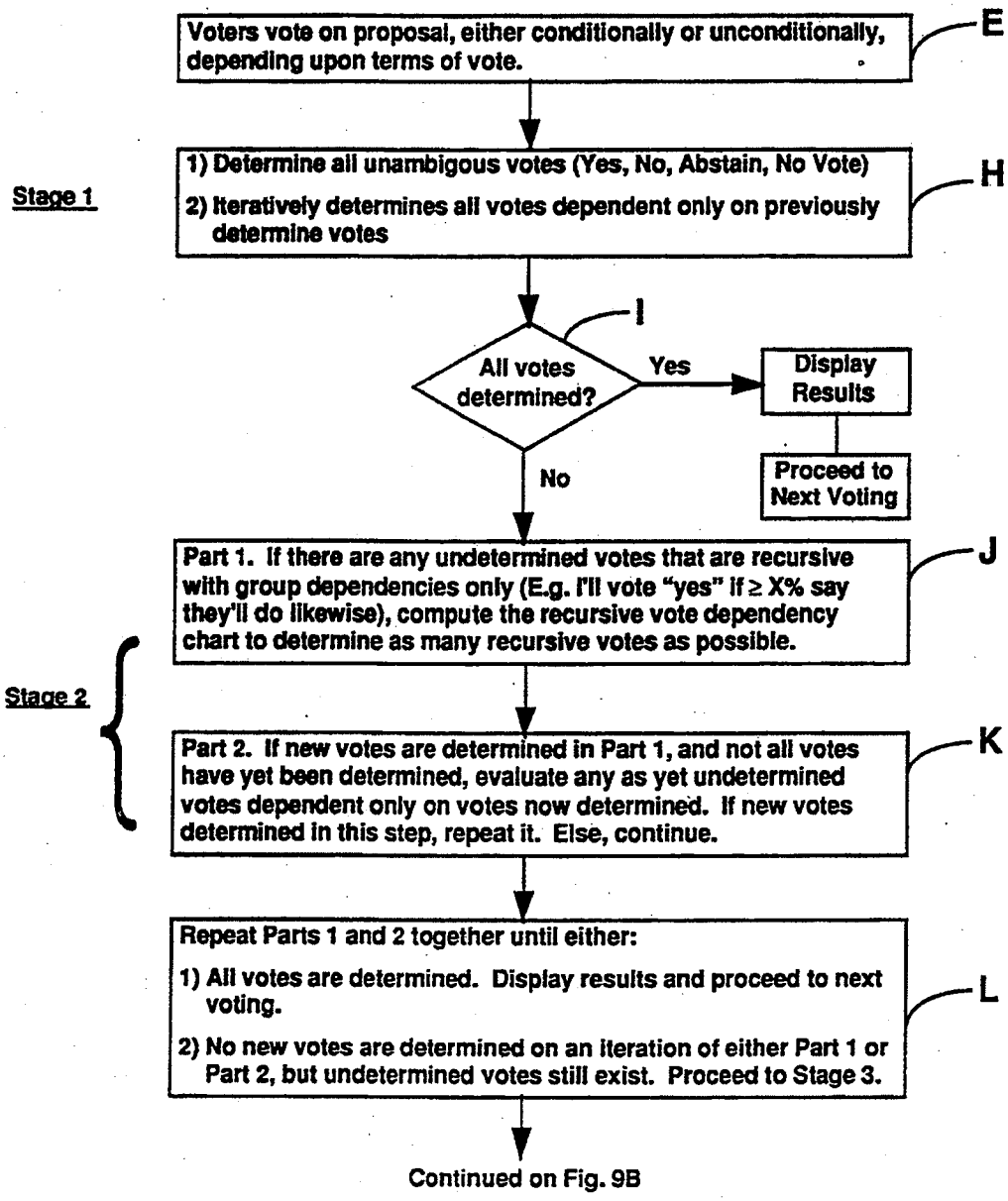

Voters vote on proposal, either conditionally or unconditionally, depending upon terms of vote.
E
Stage 1
1) Determine all unambigous votes (Yes, No, Abstain, No Vote)
2) Iteratively determines all votes dependent only on previously determine votes
H
I
All votes determined?
Yes
Display Results
Proceed to Next Voting
No
Part 1. If there are any undetermined votes that are recursive with group dependencies only (E.g. I'll vote "yes" if ≥ X% say they'll do likewise), compute the recursive vote dependency chart to determine as many recursive votes as possible.
J
Stage 2
Part 2. If new votes are determined in Part 1, and not all votes have yet been determined, evaluate any as yet undetermined votes dependent only on votes now determined. If new votes determined in this step, repeat it. Else, continue.
K
Repeat Parts 1 and 2 together until either:
1) All votes are determined. Display results and proceed to next voting.
2) No new votes are determined on an iteration of either Part 1 or Part 2, but undetermined votes still exist. Proceed to Stage 3.
L
Continued on Fig. 9B

Fig. 9A

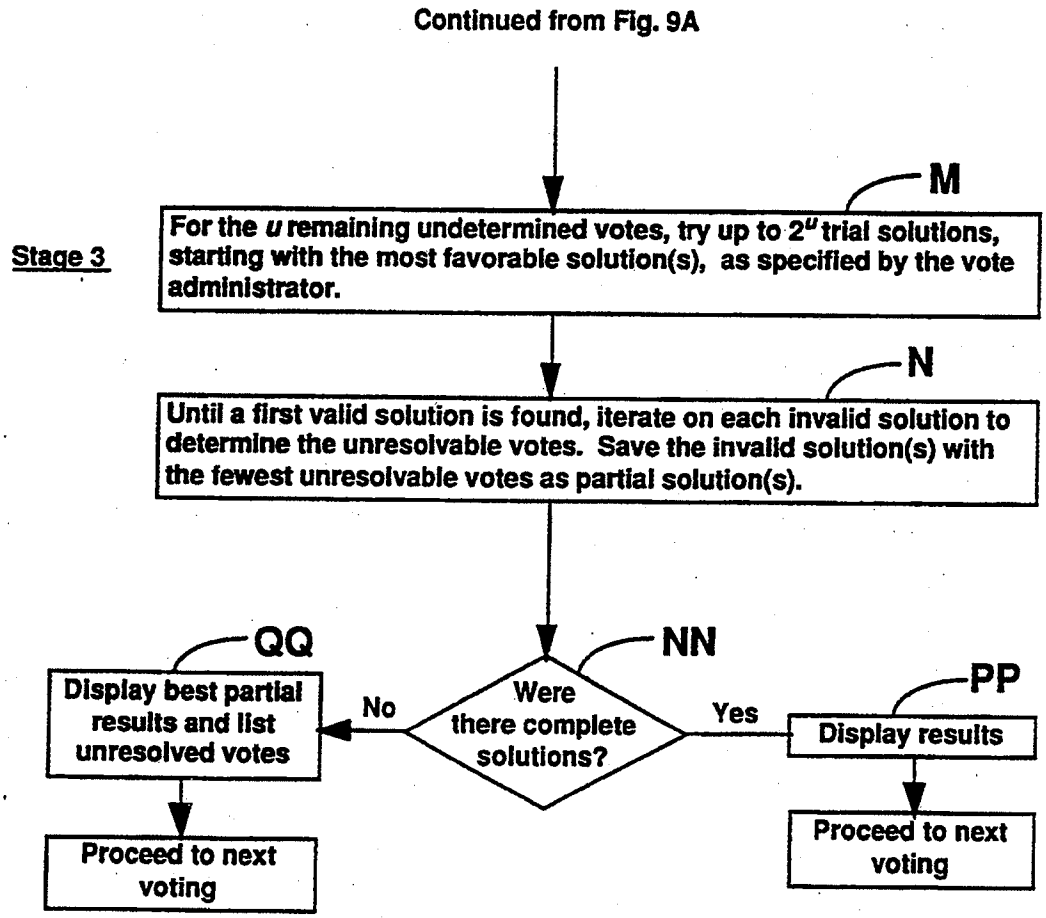

Continued from Fig. 9A
M
Stage 3
For the u remaining undetermined votes, try up to 2^u trial solutions, starting with the most favorable solution(s), as specified by the vote administrator.
N
Until a first valid solution is found, iterate on each invalid solution to determine the unresolvable votes. Save the invalid solution(s) with the fewest unresolvable votes as partial solution(s).
NN
Were there complete solutions?
No
Yes
QQ
Display best partial results and list unresolved votes
Proceed to next voting
PP
Display results
Proceed to next voting

Fig. 9B

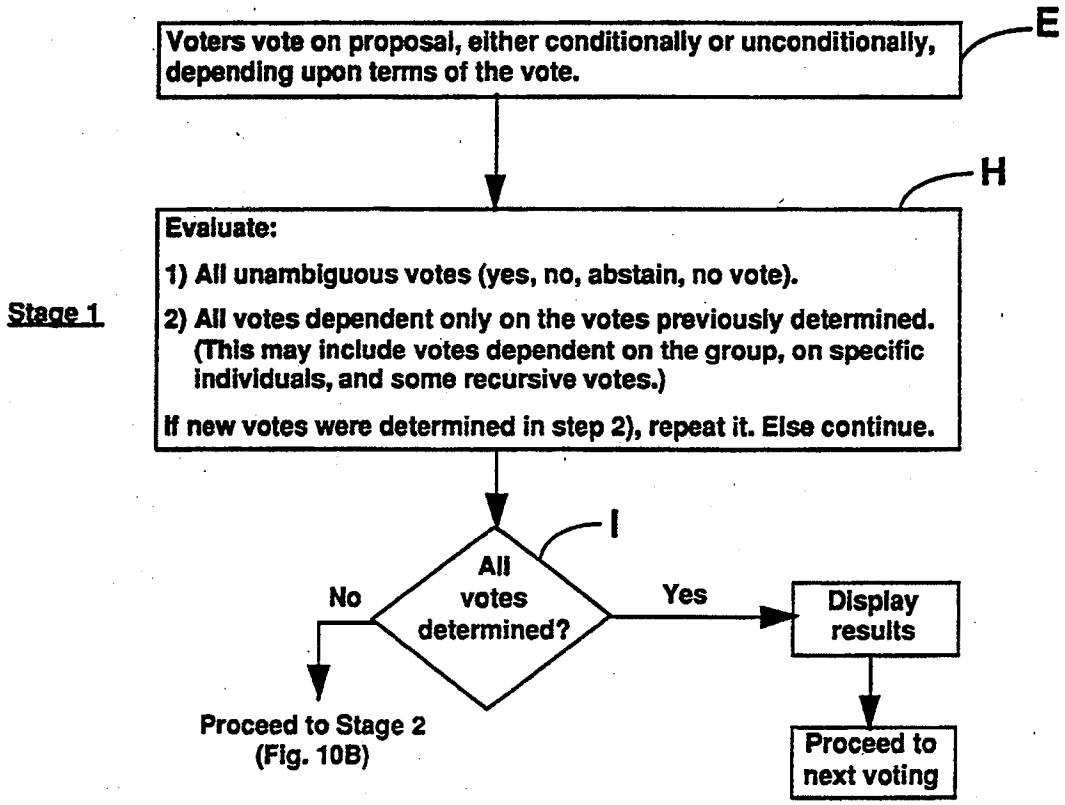

Voters vote on proposal, either conditionally or unconditionally, depending upon terms of the vote.
E
H
Evaluate:
1) All unambiguous votes (yes, no, abstain, no vote).
2) All votes dependent only on the votes previously determined. (This may include votes dependent on the group, on specific individuals, and some recursive votes.)
If new votes were determined in step 2), repeat it. Else continue.
Stage 1
I
All votes determined?
No
Yes
Display results
Proceed to Stage 2 (Fig. 10B)
Proceed to next voting

Fig. 10A

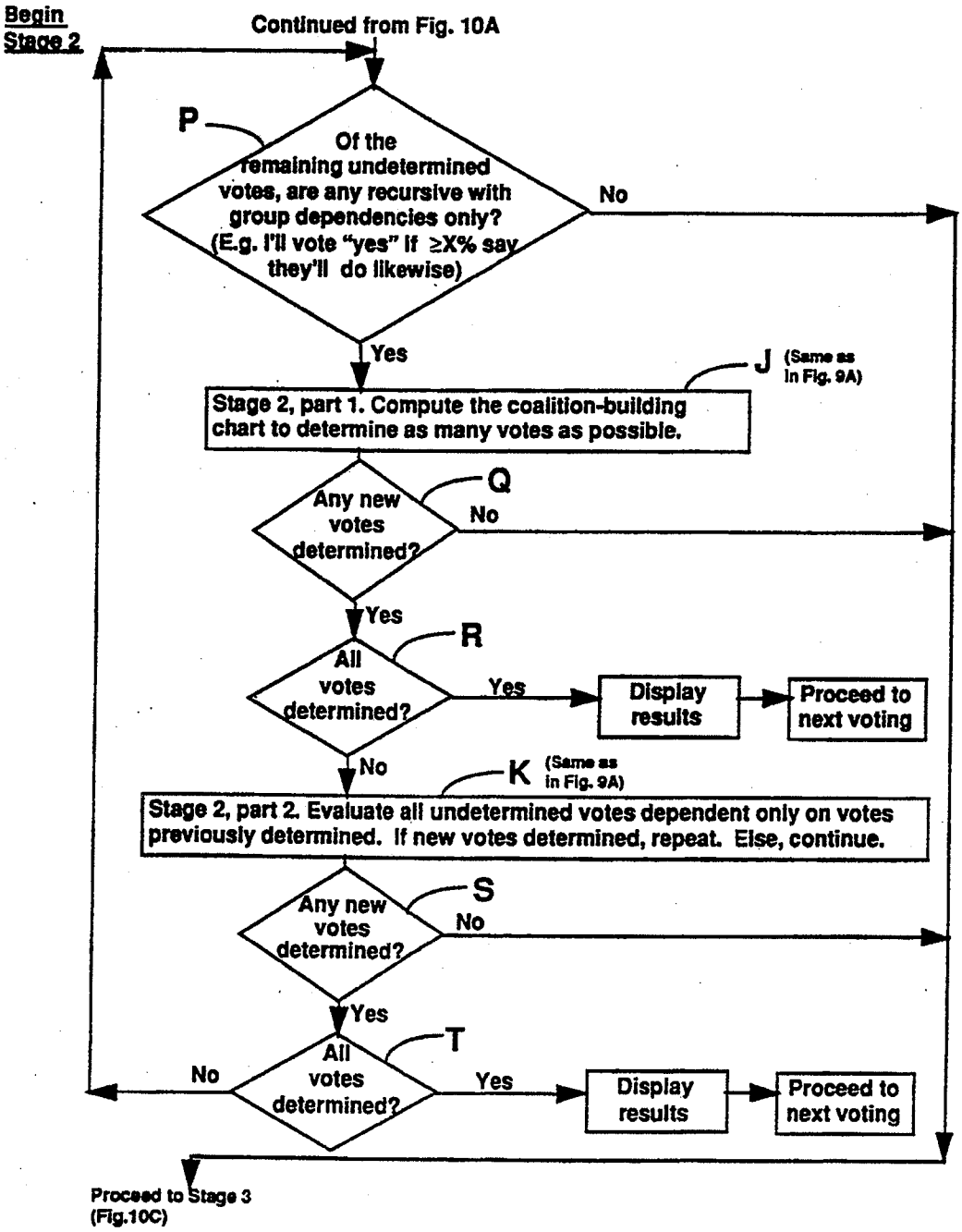

Begin Stage 2
Continued from Fig. 10A
P
Of the remaining undetermined votes, are any recursive with group dependencies only? (E.g. I'll vote "yes" if ≥X% say they'll do likewise)
No
Yes
J
(Same as In Fig. 9A)
Stage 2, part 1. Compute the coalition-building chart to determine as many votes as possible.
Q
Any new votes determined?
No
Yes
R
All votes determined?
Yes
Display results
Proceed to next voting
No
K
(Same as in Fig. 9A)
Stage 2, part 2. Evaluate all undetermined votes dependent only on votes previously determined. If new votes determined, repeat. Else, continue.
S
Any new votes determined?
No
Yes
T
All votes determined?
No
Yes
Display results
Proceed to next voting
Proceed to Stage 3 (Fig.10C)

Fig. 10B

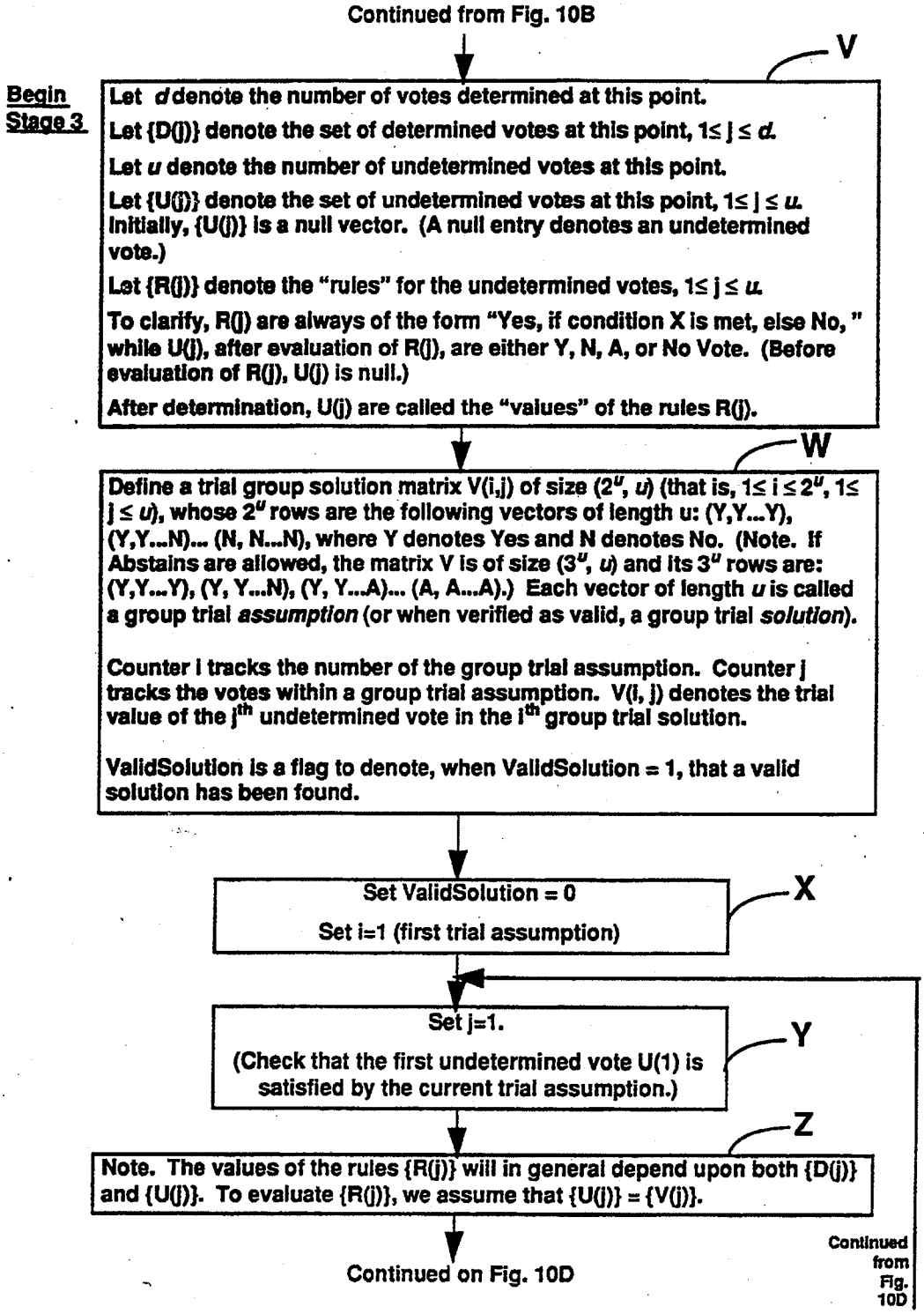

Continued from Fig. 10B
V
Begin Stage 3
Let d denote the number of votes determined at this point.
Let {D(j)} denote the set of determined votes at this point, 1≤ j ≤ d.
Let u denote the number of undetermined votes at this point.
Let {U(j)} denote the set of undetermined votes at this point, 1≤ j ≤ u. Initially, {U(j)} is a null vector. (A null entry denotes an undetermined vote.)
Let {R(j)} denote the "rules" for the undetermined votes, 1≤ j ≤ u.
To clarify, R(j) are always of the form "Yes, if condition X is met, else No," while U(j), after evaluation of R(j), are either Y, N, A, or No Vote. (Before evaluation of R(j), U(j) is null.)
After determination, U(j) are called the "values" of the rules R(j).
W
Define a trial group solution matrix V(i,j) of size (2^u, u) (that is, 1≤ i ≤ 2^u, 1≤ j ≤ u), whose 2^u rows are the following vectors of length u: (Y,Y...Y), (Y,Y...N)... (N, N...N), where Y denotes Yes and N denotes No. (Note. If Abstains are allowed, the matrix V is of size (3^u, u) and its 3^u rows are: (Y,Y...Y), (Y, Y...N), (Y, Y...A)... (A, A...A).) Each vector of length u is called a group trial assumption (or when verified as valid, a group trial solution).
Counter i tracks the number of the group trial assumption. Counter j tracks the votes within a group trial assumption. V(i, j) denotes the trial value of the j^{th} undetermined vote in the i^{th} group trial solution.
ValidSolution is a flag to denote, when ValidSolution = 1, that a valid solution has been found.
X
Set ValidSolution = 0
Set i=1 (first trial assumption)
Y
Set j=1.
(Check that the first undetermined vote U(1) is satisfied by the current trial assumption.)
Z
Note. The values of the rules {R(j)} will in general depend upon both {D(j)} and {U(j)}. To evaluate {R(j)}, we assume that {U(j)} = {V(j)}.
Continued on Fig. 10D
Continued from Fig. 10D

Fig. 10C

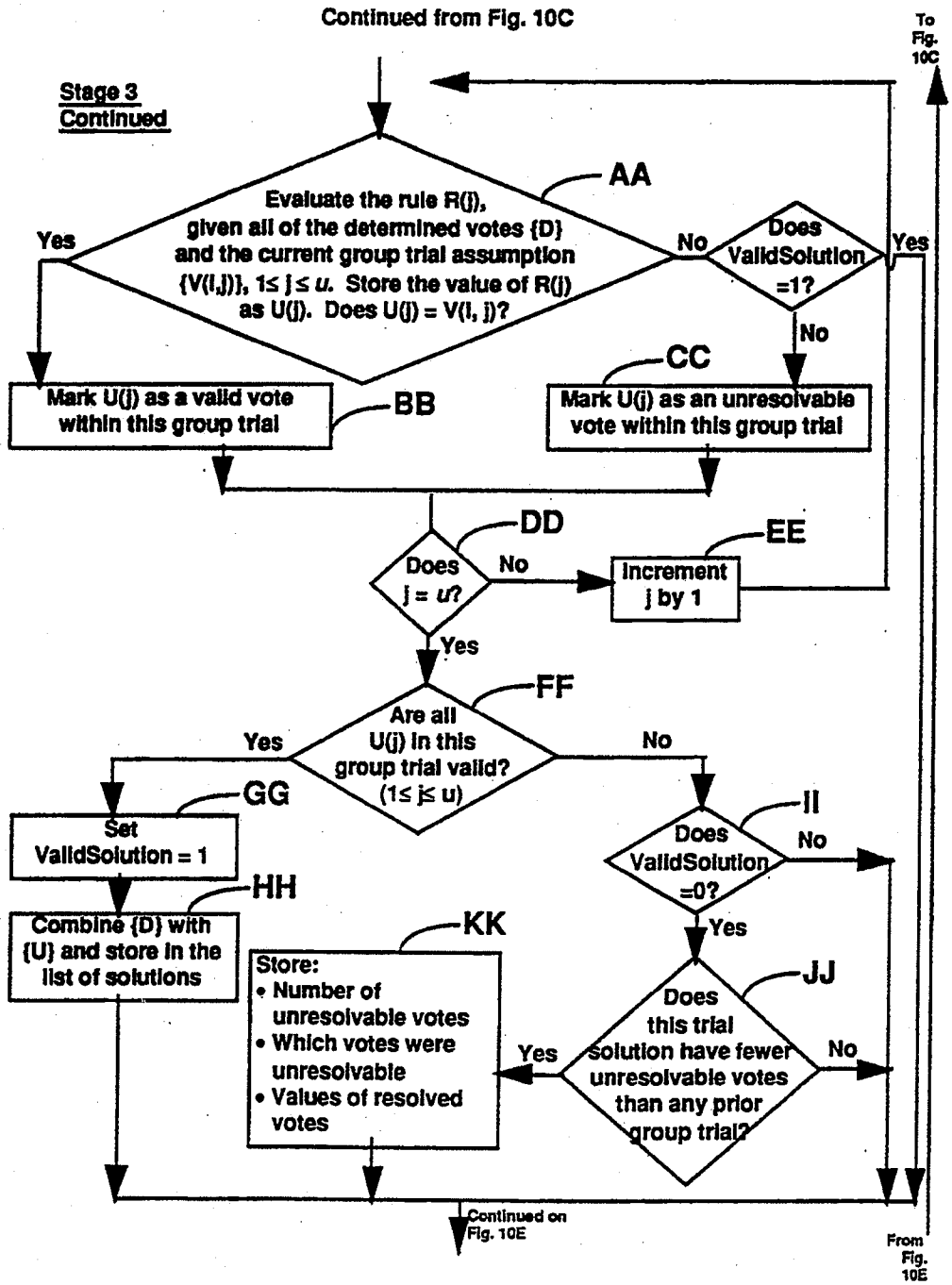

Continued from Fig. 10C
To Fig. 10C
Stage 3 Continued
AA
Evaluate the rule R(j), given all of the determined votes {D} and the current group trial assumption {V(I,j)}, 1≤ j ≤ u. Store the value of R(j) as U(j). Does U(j) = V(I, j)?
Yes
No
Does ValidSolution =1?
Yes
No
CC
Mark U(j) as an unresolvable vote within this group trial
BB
Mark U(j) as a valid vote within this group trial
DD
Does j = u?
No
EE
Increment j by 1
Yes
FF
Are all U(j) in this group trial valid? (1≤ j≤ u)
Yes
No
GG
Set ValidSolution = 1
HH
Combine {D} with {U} and store in the list of solutions
KK
Store:
• Number of unresolvable votes
• Which votes were unresolvable
• Values of resolved votes
II
Does ValidSolution =0?
No
Yes
JJ
Does this trial solution have fewer unresolvable votes than any prior group trial?
Yes
No
Continued on Fig. 10E
From Fig. 10E

Fig. 10D

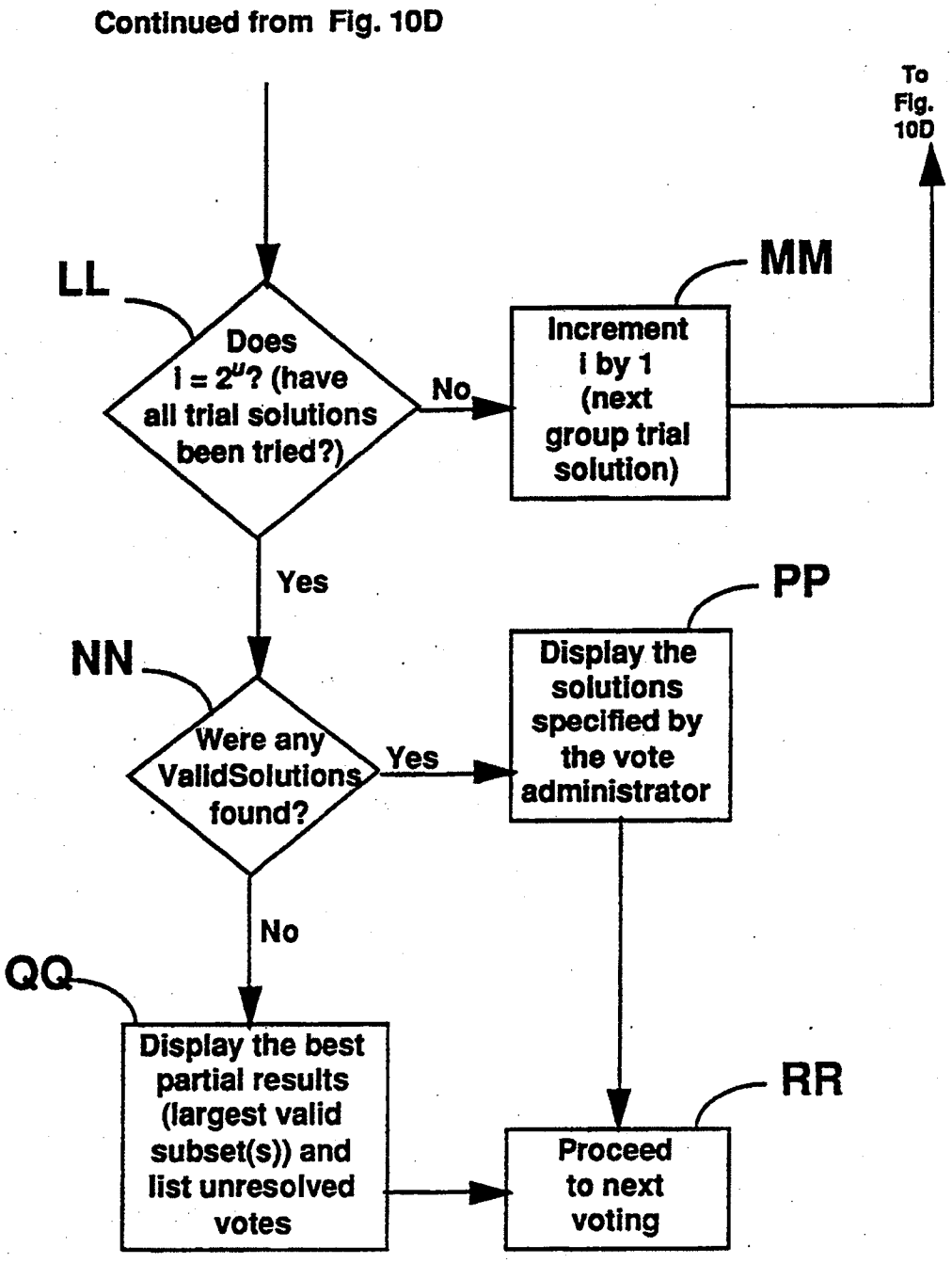

Continued from Fig. 10D
To Fig. 10D
LL
Does i = 2^u? (have all trial solutions been tried?)
No
MM
Increment i by 1 (next group trial solution)
Yes
NN
Were any ValidSolutions found?
Yes
PP
Display the solutions specified by the vote administrator
No
QQ
Display the best partial results (largest valid subset(s)) and list unresolved votes
RR
Proceed to next voting

Fig. 10E

(New) Voter Number:	1	2	3	4	5
y = number of supporters in group required for this person to support proposal:	1	2	?	2	4

Fig. 11

(New) Voter Number:	1	2	4	5	3
x:	1	2	3	4	5
y = number of supporters in group required for this person to support proposal:	1	2	2	4	?

Fig. 13

Step: / Vote:	Stage 1	Stage 2 Part 1	Stage 2 Part 2	Stage 2 Part 1 (Pass 2)	Stage 2 Part 2 (Pass 2)	Stage 3
#1 - Yes	Y	Y	Y	Y	Y	Y
#2 - Yes if and only if #1 votes yes	Y	Y	Y	Y	Y	Y
#3 - Yes if and only if ≥3 vote yes		Y	Y	Y	Y	Y
#4 - Yes if and only if #3 votes yes			Y	Y	Y	Y
#5 - Yes if and only if ≥5 vote yes				Y	Y	Y
#6 - Yes if and only if #5 votes yes					Y	Y
#7 - Yes if and only if 8 vote yes						N
#8 - No if #7 votes yes: else yes						Y

Fig. 12

Group trial assumption: / Vote:	1	2	3	4
#7 - Yes if and only if 8 vote yes, else no	Y	Y	N	N
Test:	✔	X	✔	✔
#8 - No if #7 votes yes: else yes	Y	N	Y	N
Test:	X	✔	✔	X

Fig. 14

	Best Trial			
Voter	1	2	3	4
#1	Y	-	-	N
#2	-	N	Y	-
#3	Y	Y	N	N
#4	Y	Y	N	N
#5	Y	Y	N	N

Fig. 15

	Best Partial Solution			
Voter	1	2	3	4
#1	-	-	-	-
#2	-	-	-	-
#3	-	-	-	-
#4	-	-	-	-
#5	-	-	-	-

Fig. 16

	Best Trial							
Voter	1	2	3	4	5	6	7	8
#1	Y	Y	-	-	-	-	N	N
#2	-	-	N	N	Y	Y	-	-
#3	Y	N	Y	N	Y	N	Y	N
#4	Y	N	Y	N	Y	N	Y	N

Fig. 17

COMPUTER NETWORK BASED CONDITIONAL VOTING SYSTEM

COPYRIGHT NOTICE

A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure as it appears in the Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

BACKGROUND OF THE INVENTION

The present invention relates generally to computer based voting applications, and more specifically, to a computer network based conditional voting system.

Systems for voting have existed since human beings started counting raised hands. This ancient approach did not allow for secrecy, and required that everyone in a group voting had to be at the same place at the same time. Later, the secret ballot and ballot box provided secrecy and freedom from voting at a fixed time, but still required that voters congregate at a specific place. Computer networks that reach individuals wherever they may be, through desktop, portable and hand-held input/output devices (e.g. keyboards and displays), later allowed votes to be cast by voters anywhere, without the need to congregate in one place.

All of these systems, including the modern, computer based ones, make use of very simple ballots. These ballots offer voters a limited choice, typically of one or more of the following: 1) vote Yes or No, and sometimes Indifferent and/or Abstain; 2) select one or more of multiple choices from a list; 3) write-in a desired selection; or 4) prioritize a list of alternatives.

These conventional alternatives can limit the ability of a group to make the best decision, or limit the voters from expressing their true preferences. Specifically, the conventional systems do not allow users' votes to be conditional on the votes of other members of the group. The following examples illustrate a few of the shortcomings of conventional voting systems.

Example 1. Person A may not be well informed on the issue, but knows that person B is, and has a high degree of confidence in person B's judgment. Person A may therefore wish to vote "the same as B votes", whether or not A knows what B's vote is. In traditional voting systems, the only way A can vote with B is to consult with B before the vote takes place to find out how B is voting. But if B should change his vote at the last moment, or if A has no way of contacting B, or if B has not yet decided his vote when A and B are able to discuss the matter, then A cannot guarantee that his vote is the same as B's. As a further example, A may wish to vote the way the "majority of B, C and D vote". In this example, the communication and logistical problems are three times as complicated as A merely voting with B.

Example 2. Person A's primary goal may be to support the position of a person B (perhaps the employer or spouse of person A). A may therefore choose to vote whichever way B votes on a wide variety of matters. To achieve this end with conventional voting systems, A would need to consult with B on every single matter, a time-consuming and perhaps impossible requirement.

Example 3. Persons A and B have agreed to trade votes on different issues. On issues 1, 3, 5, and 7, A will vote the same way as B. On issues 2, 4, 6, and 8, B will vote the same way as A. Again, to achieve this end with conventional voting systems, extensive and time-consuming coordination between A and B would be required.

Example 4. Person A's primary goal is to support the majority's view. Person A may therefore choose to vote whichever way the majority votes. To achieve this end with conventional voting systems, A must either guess or conduct a poll of other voters before the vote, either of which could be inaccurate or could change, to assess the majority's vote before the voting takes place.

Example 5. Person A does not particularly support an issue, but would vote in favor of it if all of persons B, G, M, P, S, W, and Z voted in favor of it. To achieve this end with conventional voting systems, would require contacting all of those individuals before the voting took place. (This example is similar to the majority of B, C and D in example 1 above.)

Example 6. The cost per person of a proposed shared asset, such as a new road or public library, is inversely proportional to the number of persons who help fund the proposed asset. Person A likes the proposal, whose overall cost is $10,000, but is only willing or able to pay up to $200 for it. There are 100 people in the group; those who support the proposal will share its cost equally. Person A would therefore vote in favor of the proposal if and only if at least 50% of the group (any 50 out of 100 people) ended up supporting the proposal ($10,000/50=$200). Each of the other 99 members of the group similarly have their own budget limitations, for example, person B is willing to pay no more than $150, and person C, no more than $125. To identify who is in the supporting group, and whether a solution is even possible, is a complex process with conventional voting systems.

Example 6 above is a case of what is more generally called the "common goods" problem. Conventional voting systems are particularly inadequate for these problems. "Common goods", such as public parks, libraries, a clean environment, labor unions, lighthouses, fire departments, or a counter-attack on a belligerent aggressor nation, are beneficial to all, but all have some cost. "Common goods" can be abused by "free riders". A "free rider" is someone or something that enjoys the benefit of the common good without helping to pay for it.

With conventional voting systems, it is often difficult to get people to pay for common goods. There is an incentive for people to wait until others pay for the goods, and then enjoy it as free riders. Consequently, beneficial measures are often postponed or not taken while people or countries wait for others to act. People need the ability to say: "I support this measure if and only if 'X' percentage or more of the group will support it," or "I support this measure if all of persons A, B, C, D and E support it". Different members of the group will have different preferences. One person may require 80% of the group's support to support the measure; someone else may require only 50%; someone else, 90%.

SUMMARY OF THE INVENTION

According to the invention, a method and apparatus is provided for a computer network based conditional voting system. The system is used by two or more persons to arrive at a decision and allows the users to vote

either unconditionally (i.e., yes, no, or abstain) or conditionally on the votes of others within the voting group. Conditional votes can be dependent upon the votes of specific individuals, a specific group of individuals, a non-specific subset of the group, the group as a whole, upon independent events, or any combination of the above. Votes need not be weighted equally.

The preferred embodiment uses a vote processor, a voting administrator, and one or more voting units. The vote processor is a computer which coordinates the overall voting process including soliciting and accepting input from the voting units and then tabulating and displaying the results. The vote administrator inputs or approves the vote proposal and terms of the vote into the vote processor. The voting units, connected to the network by any means, are of various design including desktop mountable, handheld, and mobile. In the preferred embodiment the voting units' sole task is providing input/output (I/O) for voting while in an alternative embodiment, the voting units are general purpose devices with multiple functions, only one of which is voting.

In use, one or more parties submit and/or modify a proposal requiring a vote. After the terms and conditions of the vote are stated by the vote administrator, input from each voter is solicited and accepted through the individual voting units. The voting may be either unconditional or conditional. The terms and conditions expressed by the vote administrator determine what form the conditional input can take.

In the preferred embodiment, all input must be submitted by a first deadline. The vote processor then evaluates the conditional votes according to the terms of the vote, and tabulates the results. The results are displayed to the group of voters at which time a second vote is allowed. The vote processor cycles through the process of soliciting, accepting, evaluating, processing, and displaying the results until a final deadline is reached.

Reference to the remaining portions of the specification and the drawings realize a further understanding of the nature and advantages of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** illustrates the hierarchical configuration of the present invention;

FIG. **2** illustrates the peer-peer configuration of the present invention;

FIG. **3** is a high level flowchart illustrating the operation of the computer network based conditional voting system;

FIG. **4** is a table illustrating three major types of votes;

FIG. **5** is a consensus building chart showing the evolution of a proposal over time;

FIG. **6** is a table showing the number of proposal supporters required per voter;

FIG. **7** is a re-ordering of the information from the table of FIG. **6**, ordering the voters by the number of proposal supporters required to support the proposal;

FIG. **8** illustrates a consensus-building chart;

FIGS. **9**A–B are intermediate level flowcharts showing the three stages of vote processing;

FIGS. **10**A–E are detailed flowcharts showing the three stages of vote processing;

FIG. **11** illustrates a table showing the votes before re-ordering;

FIG. **12** illustrates the table of FIG. **11** after re-ordering;

FIG. **13** illustrates a Stage 3 example in which the vote determination is done by stage;

FIG. **14** illustrates a Stage 3 example of testing group trials;

FIG. **15** illustrates a Stage 3 example of best group trials;

FIG. **16** illustrates a Stage 3 example showing no complete or partial solutions; and

FIG. **17** illustrates a Stage 3 example where only partial solutions are possible.

DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

FIGS. **1** and **2** show two different configurations of the conditional voting system disclosed in the present invention. FIG. **1** shows a "star" configuration; FIG. **2** shows a "peer-peer" configuration. Any combination of these configurations is also possible. In FIG. **1**, the voting units **14**, **16**, **18**, and **20** and a vote administrator system **26** are networked to one or more vote processors **12**. The vote processor(s) are one or more computers on which processing of votes take place. The vote processor(s) coordinate the voting process, solicit and accept input from the voting units, tabulate results, and feed back information to the voting units. If the network does not employ such a computer, as in FIG. **2**, one or more of the voting units can handle these functions.

The preferred embodiment of the computer program for performing the vote processor functions is given in the Appendix incorporated herein. Details of the specific functions of this program are described in the following specification.

The voting unit may be a desktop unit **14**, a portable, handheld unit **16**, a mobile unit **18** for moving vehicle, water, air or space craft, or it may be embedded in a computer **20**. Any number of voting units of any type are allowed. Each voting unit may either be a dedicated device, with the sole task of providing I/O for voting, or it may be a general purpose device (such as a personal computer or intelligent TV set) with multiple functions, one of which is voting. Each voting unit includes a keyboard, keypad or similar data entry device for vote input, and an information display for output.

Each voting unit is typically used by an independent decision-maker. This decision-maker may either be a person, a group of people voting as one, a computer program, or a group of computer programs acting as a single decision-making entity. It is also possible for multiple persons or programs to share the same voting unit and act as independent decision-makers. In this case, the voting system must be able to accept and recognize the input of multiple voters from that voting unit.

If a voting unit is to be used by one or more persons, the unit requires a keyboard, keypad, or other data entry device for vote input, and an information display for output. If a voting unit is to be used by one or more computer programs, the unit requires a programming interface from which the unit can accept input and to which it can write output.

The vote administrator may be either a person or a program. If it is a person, vote administrator system **26** must include a keyboard or other data entry device and a display, to allow the vote administrator to input the vote proposal and terms of the vote to the system. If the vote administrator is a program, a programming inter-

face is required between the program and the voting system for the same purpose. If the network does not employ a separate vote administrator system, as in FIG. 2, one of the voting units may perform this function.

The network 22 (FIG. 1) or the network 24 (FIG. 2) may be based on any form of local or wide area network, including cable, leased lines, switched lines, wireless, or any combination thereof. The network may be shared by other devices and applications, or dedicated to the voting system.

FIG. 3 is a flowchart of the preferred embodiment of the invention. In this case voting begins when an individual, the proposal originator, develops one or more vote proposals. A vote proposal may take many forms. It may be able to be voted on affirmatively or negatively, or it may contain multiple alternatives that can be prioritized, that is, ranked, by voters. A vote administrator is a person or program charged with specifying terms and conditions of a voting. The vote administrator may be the same as, or different from, the proposal originator. Either the proposal originator or the vote administrator must enter the proposal into the system in electronic form (step A). In the preferred embodiment the proposal is entered by keyboard. If the proposal is entered by the originator, the system makes it available electronically to the vote administrator, for example on a computer screen. Either before or after proposals are submitted, the vote administrator specifies the terms and conditions for the votes (step B), such as who may vote, voting deadline(s), and constraints, if any, on allowed vote types. The voting system then notifies members of the group through the voting units or through other means that there are one or more proposals to be voted on (step C).

Either at a convenient time or (if so dictated by the terms and conditions of the voting) a specified time, each voter reviews the proposals along with any related remarks or justifications provided by the originator. The voters themselves can then comment on each proposal (step D), and send these comments through the network to all of the voters or any subset of the voters. Finally, the voters vote on the proposal (step E), either unconditionally (yes, no, abstain), or conditionally, depending upon what vote types were allowed by the vote administrator (see different types below). If a voter does not vote by a deadline, the voting system registers the voter as a no vote.

All votes do not have to be weighted the same. If specified by the vote administrator, some votes may be weighted differently from each other. The default weighting of a vote is 1.0. If the vote administrator weights voter x's vote by the factor W(x), $0 \leqq W(x)$, then voter x's vote will be treated as W(x) separate votes in final tabulations of all of the votes.

The system then processes the votes to compute their values (represented as signals). Depending upon the types of votes allowed by the vote terms (specified by the vote administrator) and upon the specific votes cast by the voters, the processing performed will vary. As shall be seen below, a processed vote may have either a unique computed value, multiple values, or no meaningful value (i.e., no solution). An unconditional vote always has a unique value—either yes, no or abstain—but a conditional vote may have either a unique value, multiple values, or no meaningful value. The vote terms determine, among other things, how multiple values and no meaningful values of votes are handled. For example, if the computed value of a vote is either yes or no, the terms may specify that "yes" will always be selected and presented as output. This approach can help build consensus among the voters. Or, the terms may specify that both values must be presented as output. If a vote has no meaningful computed value, the terms may specify that this fact be presented as output, or they may specify that the voter who casts that vote change his or her vote. Votes that have multiple computed values are called herein indeterminate. Votes that have no meaningful values or solutions are called herein unresolvable.

The output or results provided by the system (Step F) may take any of several forms. If all of the votes have unique computed values, the results may simply be a total of the number of yeses, nos, abstains, and no votes. If some of the votes have multiple computed values, the terms will specify which value or values of these votes will be selected. These values are then included in a tally or tallies of the values of the votes with unique values. For example, suppose that the computed value of vote #1 is uniquely yes, the computed value of vote #2 is uniquely yes, and the computed values of votes #3 and #4 are either (yes and yes), or (no and no). If the terms specify that the value "yes" is always chosen for a vote with multiple computed values (assuming that one of the computed values is yes), then the tally in this case would be 4 yeses and no nos. If, on the other hand, the terms specify that all computed values of a vote must be presented, then two separate tallies—4 yeses and no nos, and 2 yeses and 2 nos—would be reported. The terms may further specify that the set or list of all computed values by voter be reported. With the first set of terms above, there would be a single set or list. With the second set of terms above, there would be two such sets or lists. If some of the votes have no meaningful solutions, they would be omitted from the tallies of yeses, nos, and abstains.

The output or results may take other forms as well. Results may include a listing of the conditional votes themselves, or as described below, static or animated graphics that show the degree of consensus in the group, either at a point in time or as it changes over time. These results may further help the group move toward consensus. Step F is divided into three (3) stages, which are illustrated in FIGS. 9A–B and 10A–E.

Many votings are iterative (step G), that is, voters cast their votes, see the results, perhaps modify their votes and comments, and vote again. The voting system again tabulates and presents the results. This process is repeated until a particular result or deadline is reached.

The present invention is capable of processing any form of vote which is conditioned on the vote of one or more members of the group. FIG. 4 is a table illustrating three categories of votes. Type 1 votes are unconditional votes with no dependencies: yes, no, abstain, no vote. Types 2 and 3 are conditional.

Type 2 votes have group dependencies (conditions) only; they are dependent only on the voting of the group as a whole (e.g., "I vote yes if and only if 50% or more of the group votes yes"), not of the votes of specific individuals. As far as group dependencies are concerned, individual voters are indistinguishable from one another. Type 2 votes are especially important for moving the group towards consensus.

For convenience, each voter can express a group condition in terms of the total number of voters, including himself/herself, or in terms of the rest of the voters, excluding himself/herself. A group condition can be

also expressed either in terms of percentages (e.g., 50% of the group) or in units (e.g., 10 or more votes).

Type 3 votes are dependent on the votes of specific individuals in the group (e.g., "I vote yes if and only if A, B and C vote yes"). Type 3 votes may also have group dependencies (e.g., "I vote yes if and only if A votes yes, and if 50% or more of the group votes yes"). Type 3 votes, in effect, may be conditional on any logical statements L and L' about the voters. In the simple case of yes, no and abstain votes, the general form of a type 3 vote is: "Yes, if L is true; else no, if L' is true; else abstain." In the more complex case of prioritized or ranked lists of alternatives, the general form of a type 3 vote is: "Rank order #1, if L is true; rank order #2, if L'' is true; rank order #3, if L''' is true; rank order #4, if L'''' is true, etc."

Other examples of type 3 votes are: (i) voting the same as another person's vote; (ii) voting the opposite of another person's vote; (iii) voting the way the majority of parties x, y, and z vote; and (iv) voting yes if at least 50% of the group, including x, y, and z vote yes.

The conditions of a vote may themselves be conditional or unconditional, as in the following two examples:

i) Voter #1 votes yes if and only if: voter #2 votes yes unconditionally (an unconditional condition).

ii) Voter #1 votes yes if and only if: voter #2 votes yes, conditionally or unconditionally (a conditional condition).

Assuming that voter #2 votes yes if and only if voter #3 votes yes, conditionally or unconditionally, and voter #3 votes yes unconditionally, then in the first case above voter #1 would vote no. This is because in the first case #1's vote was conditioned on 2 vote being unconditional, which it wasn't. In the second case voter #1's vote would be yes since its condition allows for voter #2's yes vote to be either conditional (which it is) or unconditional.

In the preferred embodiment, if a condition is not specific as to whether it is conditional or unconditional, it is assumed to be conditional. For example, in "Voter A votes yes if B votes yes", it is assumed that what is important to A is that the final determined value of B's vote is yes, rather than how B arrived at it. If it is important to A that B vote yes unconditionally, this should be specified in A's condition.

Conditional conditions also apply to votes with only group dependencies. For example:

i) Voter #1 votes yes if and only if 60% or more vote: yes unconditionally (an unconditional condition).

ii) Voter #1 votes yes if and only if 60% or more vote yes (conditionally or unconditionally).

Assuming that voter #1 is one of three voters, and voters #2 and #3 vote as before, in the first case voter #1's vote would be no, since only #3 voted yes unconditionally. One vote out of three is less than the required 60%. In the second case, voter #1's vote would be yes. Not only do voters #2's and #3's votes meet #1's condition, but #1's own yes vote helps meet #1's condition. #1's vote is called recursive with a group dependency only.

In this case, recursion did not affect vote #1. But in other cases, it may. For example:

Voter #1 votes yes if two or more vote yes,
Voter #2 votes yes if two or more vote yes.
Voter #3 votes no (unconditionally).

In this case, vote #1 relies on both vote #2 and itself to meet its condition; vote #2, in turn, relies on both vote #1 and itself. Note that in this case #1's and #2's votes could be either both yes or both no. These cases are resolvable but are indeterminate, that is, they have multiple solutions (determined in stage 3 of the voting system). In general, allowing voters to vote the same way as each other can lead to multiple solutions. In the simplest case, if A votes the same way as B, and B votes the same way as A, the two votes could be either Yes-Yes or No-No.

The terms set by the vote administrator determine whether the system presents or reports all or a subset of the multiple solutions, when they arise. In the interest of consensus, the default assumption where a group of votes has multiple solutions is usually the one with the most yeses.

The vote administrator may specify any of the following output alternatives: i) present all solutions; ii) present all solutions that meet certain criteria, such as all solutions with three or more yeses; or iii) present only those solutions with either the most yeses or the most nos; or iv) present an "average" of all solutions. In addition to any of these four alternatives, the system can recommend to voters whose votes cause the multiple solutions how their votes can be modified to eliminate multiple solutions.

A different problem is encountered in the following scenario:

Voter #1 votes yes if #2 votes yes; else, no.
Voter #2 votes no if #1 votes yes; else, yes.

Restating this example more simply, A votes the same as B, and B votes the opposite of A. There is no solution. Allowing voters to vote opposite the way of others can lead to this result. These votes are called unresolvable.

In these cases, the system reports whose votes contain no solution, either to only the individuals casting those votes or to the group as a whole, depending upon the terms of the vote. One or both of the voters need to change their votes to make a solution possible. A partial solution is a subset of all of the votes that have a solution. When there are unresolvable votes, the voting system identifies the partial solutions with the most votes, and identifies the unresolvable votes.

Combinations of the above solution types are quite possible. A group of votes may contain some votes with multiple values and others that are unresolvable. The voting system can handle any set of conditions on the group as a whole or on individual members of the group, no matter how complex or intertwined.

The system can determine which vote or votes are most or least critical in obtaining a particular result. One way to do this is by tabulating the number of times particular votes are referenced as conditions in others' votes. In addition, the number of times a vote is used in conditions negatively may be subtracted from the number of times the vote is used in conditions positively to arrive at the vote's net positive impact on a particular result.

Certain votes with only group dependencies (no individual voter dependencies) can help build consensus and overcome the common goods/free rider problem. Called consensus building votes, they allow a voter to vote in favor of a proposal, such as a common goods proposal, if and only if a specified percentage of the entire group supports it. For example: "I will support the building of a public park in our neighborhood (and contribute my share of the cost) if and only if 80% of the voters similarly support it." If at least 80% of the

group vote similarly, the park will be approved. Those voters will have the assurance that at least 80% of the group will contribute their share.

The general form of the consensus building vote type is; "I vote yes if greater than or equal to x% of the group vote yes; else, I vote no." Another form of the consensus building vote type, most appropriate if all members of the group are casting votes that are either of this type or unconditional is; "I vote yes if greater than or equal to x% of the group vote yes, if greater than or equal to y% vote yes; else I vote no," where $0 \leqq x \leqq 100$ and $0 \leqq y \leqq x$.

In contrast, a form of vote which is not as effective in moving the group towards consensus is; "Yes, if $\geqq x\%$ vote yes unconditionally; else, no."

The consensus building vote type allows a voter to commit to what may be a risky position only if a certain number or percent of the rest of the group similarly commit, reducing the risk to all of the voters.

If all votes are either unconditional or of the consensus building type, the voting system can provide a novel graphical display of the degree of agreement or disagreement among the voters. The consensus building chart helps the group move towards and reach consensus. It may also be possible to produce the chart if some of the votes are neither unconditional nor consensus building. Even if the chart is not requested as output, the system typically performs the sorting and comparisons needed to draw the chart to value consensus building votes. If more than one solution is possible, the chart identifies the solution with the largest number of yeses.

The consensus building chart allows a group to see how close or far away it is from achieving consensus, or from achieving a coalition of a particular size. If a voting has successive iterations, the graph may vary with each iteration. In that case, the graph can be updated or played back in real time, allowing voters to review an animated history of the group's preferences as they have evolved, to visually gauge the momentum towards consensus, or to pinpoint turning points or major events in the group's dynamics. FIG. **5** illustrates a consensus building chart with three successive iterations; y_1, y_2, and y_3. The figure shows how the acceptance of the proposal has changed with each successive vote, indicating that at least some of the individual voters have modified their votes. Similarly, if y_1, y_2, and y_3 were to indicate three different proposals, then a chart looking like FIG. **5** could be used to gauge the relative acceptance of each distinct proposal.

The chart orders the voters available (x axis) to support a proposal by the total number or percent of voters they require (y axis), including themselves, to support the proposal. Where the number of voters available equals or exceeds the number of voters required, those voters' votes can be correctly determined to be yes votes. These voters are said either to support the proposal or to form a coalition favoring the proposal.

For example, a group of six voters votes as follows:

- Amy (cautiously) supports a proposal if and only if five (5) or more people in the group, including herself, support it (maximum of one dissenter).
- Bill, a strong advocate, supports the proposal in any event (no one other than himself is required for his support).
- Charlie also supports the proposal in any event.
- Dave votes in favor if at least one other person supports it.
- Ed votes in favor only if the group is unanimously in favor of it (6 supporters).
- Frank opposes the proposal under all circumstances.

FIG. **6** is a table illustrating the above information. Ordering the voters by their conditions as described above, Bill and Charlie would be ordered #1 and #2; Dave, #3; Amy, #4; Ed, #5 and Frank, #6 (FIG. **7**). If Charlie subsequently raised his requirement from one person to three, he would shift from #2 to #3, and Dave would shift from #3 to #2. In FIG. **7** this ordering is called "x".

The system then plots the number of members needed (y) to form a coalition as a function of the ordered members (x). If the number of members required (y) is less than or equal to the number of members available (x), a coalition can form. The members of the coalition are those members of the group for whom $y(x) \leqq x$ (FIG. **8**).

In general, wherever y(x) dips below the $y=x$ line, a coalition can form. If $y(x)>x$ for all values of x, no ones' conditions are met, and no coalition can form. In general, the supporting coalition will include all voters x_c whose values of x are smaller than x_{max}, where x_{max} is the largest value of x for which $y(x_{max}) \leqq x_{max}$. The supporting coalition may well include values of x_c for which $y(x_c)>x_c$. But as long as $x_c<x_{max}$, and $y(x_{max}) \leqq x_{max}$, x_c's vote can be correctly interpreted as a yes (there may be other solutions as well). In the example above, Bill, Charlie and Dave are in the coalition, because x_{Bill} and $x_{Charlie}<(x_{Dave}=x_{max})$, and because $y(x_{Dave})<x_{Dave}$.

FIGS. **9**A–B and **10**A–E detail the three stages of step F in FIG. **3** (step E from FIG. **3** is shown in FIGS. **9**A and **10**A to provide a frame of reference). FIGS. **10**A–E are a more detailed version of FIGS. **9**A–B. In brief, the purposes of the three stages are as follows:

- Stage 1 (steps H and I)—Assess all unconditional votes (type 1 votes—yes, no, abstain, no vote) and all votes directly or indirectly dependent on those unconditional votes only (some type 2 and 3 votes).
- Stage 2 (steps J through L in FIG. **9**A; steps P through T in FIG. **10**B)—Iteratively determine 1) consensus building vote types and 2) votes dependent on all previously determined votes. Repeat these two steps until no more new votes can be determined.
- Stage 3—(steps M through QQ in FIG. **9**B; steps V through RR in FIGS. **10**C, **10**D, and **10**E)—Assess any remaining votes (some type 2 and 3) using a trial solution method. Find multiple solutions wherever they apply. Until the first complete valid solution is found, determine the largest valid subset of each trial assumption. Report those complete or partial solutions specified by the terms of the vote.

Stage 1

At the beginning of stage 1 (step H), the system examines the votes in whatever order they happen to be in, and identifies the unconditional ones (yes, no, abstain, no-vote). As the system examines each vote, it also evaluates any conditional votes that have become determinable as a result of unconditional votes now determined. Votes so determined are conditional votes that are dependent only on the unconditional votes. Then the system passes through the list again and evaluates all new votes that are dependent only on the ones previously determined, either conditional or unconditional. This process is repeated until an iteration occurs on which no new votes are determined. If all votes in the group have been determined by this process (step I), the

system is finished and the results are displayed. If all votes have not been determined, we proceed to stage 2 (step J in FIG. 9A; P in FIG. 9B).

As an example, a group of 5 people vote as follows. (Note that although the vote types are indicated, the system does not need to recognize or categorize them as such.)

#1: Yes if ≧2 people vote yes, else no (type 2);
#2: Yes if #5 votes yes, else no (type 3);
#3: Yes if #4 votes yes, else no (type 3);
#4: Yes if ≧4 people vote yes, else no (type 3);
#5: Yes (type 1).

On the first pass, only #5 is determined (an unconditional yes vote). On the second pass, #2 is determined (yes). On the third pass, #1 is determined (yes). On the next pass, no new votes are determined, since both #3 and #4 depend on more than just #1, #2, and #5. #3 depends upon #4, and #4 may depend upon either itself or #3. The system proceeds to stage 2.

Alternatively, the system can first order the votes from simplest to most complex conditions. By doing so, fewer passes may be required. For example, the system could first order the above votes as follows:

New #1 (old #5): Yes;
New #2 (old #2): Yes if (new) #1 votes yes, else no;
New #3 (old #3): Yes if (new) #5 votes yes, else no;
New #4 (old #1): Yes if ≧2 people vote yes, else no;
New #5 (old #4): Yes if ≧4 people vote yes, else no.

After ordering as above, on the first pass new votes #1, #2, and #4 (old votes #5, #2, and #1, as before) would be determined. On the next pass, no new votes would be determined, and the system would proceed to stage 2.

Stage 2

At the beginning of stage 2 (J in FIG. 9A, P in FIG. 10B), the system first checks whether there are any votes not yet determined of the form:

Yes if ≧x% of the people vote yes, else no ($0 \leqq x < 100$).

If so, the system computes the consensus building chart (J in both FIGS. 9A and 10B) for as many votes as fit the chart format, to determine as many consensus building votes as possible.

In the case above, there is one vote (new #5=old #4) of the consensus building type that has not yet been determined. Using the new numbering scheme, the system creates the table shown in FIG. 11. The system then reorders the voters (FIG. 12).

Since $y(x) \leqq x$ for voters #1, #2, #4, and #5, we now know that voter #5 can also support the proposal (i.e., yes is a valid value for the #5 vote), in addition to supporters #1, #2, and #4 already determined. Note that the chart is incomplete because voter #3's vote has not yet been determined at this point. Since a new vote, #5, was determined (step Q), and since all votes have not yet been determined (step R), the system performs step K, which is identical to the second part of step H (FIGS. 9A and 10A). Step K evaluates any votes dependent only on votes previously determined. In step K, we determine #3 (yes) from the newly determined #5 vote (yes). The solution with the most number of yeses (all yeses) has now been determined. If requested, the system draws the consensus building chart and presents it to the voters. (If the voting terms specify finding other solutions as well as the one with the most number of yeses, the system will proceed with stage 3 to search for other solutions.)

Stage 3

Unless the terms of the vote specify finding solutions other than the one with the most number of yeses, stage 3 is reached only if undetermined votes remain after stages 1 and 2. Let u be the number of undetermined votes at the end of stage 2. To evaluate these votes, the system generates (step W) and tries (step X) the 2^u possible combinations of u votes: (Y,Y . . . Y); (Y,Y . . . N); . . . (N,N . . . N). (If Abstains are allowed, there are 3^u possible combinations of u votes: (Y,Y . . . Y); (Y,Y . . . A); (Y,Y . . . N); . . . (N,N . . . N).) These combinations are called group trial assumptions or, if they satisfy all of the votes' conditions, group trial solutions.

There are two nested main logic loops in stage 3, a larger outer loop and a smaller inner loop. The outer loop tests a particular group trial assumption. This outer loop starts after step X and ends at step MM (FIG. 10E), at which point counter i is incremented for the next group trial assumption. (Counter i is initialized in step X). The inner loop tests a single undetermined vote within the larger group trial. This inner loop starts at step AA and ends at step EE, at which point counter j is incremented for the next undetermined vote within the larger group trial. (Counter j is initialized in step Y). If u is the number of undetermined votes, there will be up to u circuits of the inner loop for each circuit of the outer loop.

Step AA tests whether the conditions of a particular undetermined vote are met by the current group trial assumption. If so, control passes to BB, where the vote is marked as valid. If not, control passes to CC, where the vote is marked as unresolvable for this group trial.

When all of the votes in a particular group trial have been tested, control passes to step FF, which tests whether the conditions of all of the undetermined votes have been satisfied by the given group trial assumption. If so, the flag "ValidSolution" is set to 1 (GG), and the group trial solution is stored (HH). If the conditions of any undetermined vote are not met by the group trial assumption, control passes to step II. Step II checks whether a complete valid solution has yet been found. If so (ValidSolution=1), control passes directly to step LL to check whether there are any more group trials to be tested, without saving the partial solution (partial solutions are generally of less interest than complete solutions).

If no complete valid solution has been found (ValidSolution=0), control passes from step II to step JJ, which tests whether the group trial has the same or fewer number of unresolvable votes than any prior group trial. If so, the group trial is saved in step KK before control passes to step LL.

Step LL checks whether there are any more group trials to be tested. If so, i is incremented (MM) and the next trial is tested. If not, step NN checks whether any complete solutions were found. If so, the system displays the complete solutions specified by the terms of the vote (PP). If none were complete, the system determines (as described in examples #3 and #4 below) primary and secondary unresolvable votes (QQ) and displays the best partial solution(s).

Example #1

The following example of a group of eight voters requires going through stages 1 through 3 of the voting system. To avoid multiple iterations in stage 1, the votes have been ordered in an optimal way. This does not affect the result. The group votes are as follows:

#1: Yes

#2: Yes if and only if #1 votes yes
#3: Yes if and only if 3 or more people vote yes
#4: Yes if and only if #3 votes yes
#5: Yes if and only if 5 or more people vote yes
#6: Yes if and only if #5 votes yes
#7: Yes if and only if 8 people vote yes
#8: No if #7 votes yes, else yes

As mentioned above, all conditions not explicitly described as unconditional may be either conditional or unconditional. FIG. **13** shows the step at which each vote is determined. In this figure, the bold votes (Y) indicate new votes determined in that step.

In stage 3 of this example, four group trial assumptions are considered for votes #7 and #8 (FIG. **14**): YY, YN, NY, NN. Only one of the four combinations, #7=no, #8=yes, meets both sets of constraints. The solution is unique. In FIG. **14**, a check mark (✓) indicates that the vote condition was met by the group trial assumption, while an x (X) indicates that the condition was not met.

Working through the problem, as illustrated in FIG. **14**, group trial assumption 1 is #7=Y, #8=Y. This trial satisfies the condition for vote #7 (first passes of step AA and BB)—"all 8 votes in the group are yes"—but fails the condition for vote #8 (second pass of step AA, first pass of step CC)—"no if #7 is yes". Since not all trial solutions have been tested (DD), i is incremented by 1 (EE) and the next trial solution is tried. After a similarly unsuccessful test of group trial assumption 2, group trial assumption 3 (#7=No, #8=Yes) is tried which satisfies the conditions of both voters #7 and #8. The ValidSolution flag is set equal to 1 (GG). The solution for voters #7 and #8 is combined with the results for the previously determined votes #1 through #6 (denoted {D} in step HH) and stored. After the remaining trial assumption 4 is tried (unsuccessfully), the unique solution (seven yeses, one no) is displayed as specified by the voting terms.

Example #2

This group of four votes is evaluated entirely in stage 3:

#1: Yes if and only if all four votes are yes
#2: Yes if and only if all four votes are yes
#3: Yes if and only if all four votes are yes
#4: Same as #3

The $2^4=16$ group trials evaluated are (Y,Y,Y,Y) through (N,N,N,N). Of these, (Y,Y,Y,Y) and (N,N,N,N) are the two complete valid solutions. Most likely, (Y,Y,Y,Y) would be the preferred solution.

Example #3

This set of five votes has no complete or even partial solution:

#1: vote same as #2
#2: vote opposite of #1
#3: vote same as #1
#4: vote same as #3
#5: vote as the majority of (#2, #3, and #4)

Votes #1 and #2 are opposite each other, and the remaining votes all depend on those conflicting votes. The set of votes is assessed entirely in stage 3. The $2^5=32$ group trials evaluated are (Y,Y,Y,Y,Y) through (N,N,N,N,N).

As usual, for all group trials, we test whether every vote's condition is directly satisfied by that trial. For example, all of the votes in the example except #2's are satisfied by (Y,Y,Y,Y,Y) (vote #2 should be the opposite of #1, which it is not). After testing all of the trials in this way, we identify those with the fewest number of unresolvable votes. The unresolvable votes in this set of best trials are called the primary unresolvable votes or the unresolvable kernel. In the current example, at this first step, no trials have no unresolvable votes, but four trials have only one unresolvable vote each. In each trial, the unresolvable vote is either #1 or #2. FIG. **15** shows these four trials, and identifies votes #1 and #2 as the unresolvable kernel. In this figure, the unresolvable votes are marked by a strikethrough (-), the yes votes by a Y, and the no votes by a N.

When none of the group trials produces a complete valid solution, finding the best partial solution has a second step: checking which votes are dependent on the unresolved votes. These votes are also unresolvable. In best trial 1 (FIG. **15**), for example, vote #1 is dependent on vote #2, which is unresolvable. So #1 is unresolvable. And since all of the other votes in trial 1 are dependent upon #1 and #2, they, too, are unresolvable. These votes are called secondary unresolvable votes. A similar analysis of best trials 2, 3, and 4 shows that all of the votes in FIG. **15** are unresolvable (FIG. **16**). The dash mark (-) in FIG. **16** indicates an unresolvable vote.

The system reports that votes #1 and #2 are primary unresolvable votes, and that there are no partial solutions. Either voter #1 or voter #2, or both, must change their votes so as not to be opposed to each others' votes to make even a partial solution possible.

Example #4

This set of four votes, similar to example #3, has no complete solution, but does have partial solutions. The four votes are:

#1: vote same as #2
#2: vote opposite of #1
#3: vote same as #4
#4: vote same as #3

Votes #1 and #2 are unresolvable, and votes #3 and #4 have multiple solutions. The set of votes is assessed entirely in stage 3. The $2^4=16$ group trials evaluated are (Y,Y,Y,Y) through (N,N,N,N).

No trials in this example yield complete valid solutions, but eight trials have only one unresolvable vote each, either #1 or #2. FIG. **17** shows these eight trials, and identifies votes #1 and #2 as the unresolvable kernel. Once again, the unresolvable votes are marked by a strikethrough (-), the yes votes by a Y, and the no votes by a N.

In the second step, the votes are checked for dependency on the unresolved votes, which would also be unresolvable. There are no such votes. Assuming the best result is the one with the most yeses, then the best result would be:

#1: Primary unresolvable vote (or unresolvable kernel)
#2: Primary unresolvable vote (or unresolvable kernel)
#3: Yes
#4: Yes

The system reports that there are no complete solutions, that votes #1 and #2 are primary unresolvable votes, and that the best partial solution is as shown above. Either voter #1 or voter #2, or both, must change their votes to make a complete solution possible.

Vote Administration

Before a voting can occur, the terms and conditions under which it is to take place must be set. This is the

job of vote administrator **26**. The vote administrator may be the same or different from the proposal originator. The vote administrator specifies the following:

Proposition: Statement of the proposal or question on which the group is voting.

Membership: Who is in the voting group.

Acceptance criteria: What constitutes acceptance of the proposition? Greater than 50%? Two-thirds majority? Unanimous? Any percentage can be specified. If the purpose of the vote to identify the largest coalition(s) that support the proposal, acceptance criteria are optional. If used, acceptance criteria can specify the minimum number or percent of members that must be in the coalition for the overall group to acknowledge/accept the coalition. The system will typically identify the largest coalition or coalitions (if there are more than one of equal or near equal size) that supports the proposal.

Vote iterations and deadline: A conditional voting will typically be an iterative process with feedback provided. The iterations may be either of two types:

i) Discrete—A succession of discrete, scheduled votes are conducted, usually at regular intervals, until a certain deadline is reached. Updated vote results are provided to the users immediately after each iteration. Voters may change their votes any number of times before each scheduled vote without effecting the feedback of that vote; only the vote cast at the scheduled voting time effects the feedback. Similarly, voters may change their votes any number of times in the scheduled votes before the final vote is conducted. Only the vote cast in the final voting is counted. The administrator may specify that votes be cast near simultaneously, within some narrow specified time interval.

ii) Continuous—On-going voting process during which feedback is provided continuously. Only the vote cast at the time of the deadline counts.

Allowability of modified proposals: The administrator **26** can specify whether modifications of an original proposal/question may be put forward by members of the group, and if so, under what circumstances. Modifications can be allowed only if first approved by:

i) The group administrator
ii) Some number or percent of the group
iii) The proposal originator
iv) Any combination of the above

Allowed vote types: Unconditional yes, no, abstain and no-vote comprise the simplest set of votes. If conditional votes are also allowed, they may include: 1) conditional only on the votes of people who vote unconditionally; 2) conditional only on the number of votes cast the same way, conditionally or unconditionally; 3) conditional on the votes of specific people cast the same way, conditionally or unconditionally (e.g. I will vote yes if and only if at least two of A, B and C vote yes); 4) conditional on either the number of people or specific people voting either the same way or the opposite way, with or without qualification (e.g. if A votes yes, I will vote no; if A votes no, I will vote yes); 5) any combination of the above. In addition to simple yes, no, and abstain votes with or without conditions, allowed votes may include prioritized lists of alternatives, with or without conditions.

Vote weightings: whether all votes will be weighted equally or some votes weighted differently from each other. The default weighting of a vote is 1.0. If voter x's vote is weighted by the factor $W(x)$, $0 \leqq W(x)$, then voter x's vote will be treated as $W(x)$ separate votes in final tabulations of all of the votes.

Blocking: Whether voters can "block" other voters from including the former's votes in the latter's conditions.

Displayed results: In the alternatives below, "yes" can be replaced by "no", if so specified by the vote administrator.

Unique or multiple complete solutions:

1. The best solution, if more than one solution (i.e. the one with the most number of yeses)
2. All complete valid solutions, listed in order of descending number of yeses
3. Any specified number of the top best solutions from list 2 above, listed by descending number of yeses
4. All solutions in which the yes votes are not a strict subset of the yes votes of another solution
5. Any specified number of the top best solutions from list 3 above, listed by descending number of yeses
6. The consensus building chart; with either names, name codes, or neither; presented in either numbers of voters or percent of the group
7. Any combination of the above alternatives (including alternatives based on both yeses and nos)

Partial solutions only:

i) Best partial solutions
ii) Primary unresolvable votes
iii) Secondary unresolvable votes

Handling of unresolvable votes:

i) Identified to casters of unresolvable votes only
ii) Identified to all voters

Additional information, such as the breakdown of votes by various voter categories, can also be displayed.

Output confidentiality: Whether the voters are to be fully identified, end-result identified, fully anonymous, labelled, or probabilistically anonymous.

"Fully identified" means that all votes, including the conditions upon which the votes are based, are available to everyone.

"End-result identified" means that the final result (yes, no, abstain, indeterminate, and non-vote) of everyone's vote is identified with that voter, but not the conditions the voter specified which led to the final result.

"Fully anonymous" means that only simple vote tallies—the number of yes, no, abstain, and indeterminate votes, and the number of non-voters—are made available to everyone.

"Labelled" means that voters are identified throughout the voting process by means of labels, that allow their behavior to be tracked but not the voters identified.

"Probabilistically anonymous" means that the vote administrator can specify a probability P, $0 \leqq P \leqq 1$. The vote information made available to the voters is the same as "fully anonymous" or "labelled", depending upon the choice of the administrator, with probability 1-P, and is the same as "fully identified" with probability P. The probability P may be applied either to the voting group as a whole, or to voters individually and

independently, depending upon the choice of the administrator.

The vote administrator can also specify whether any of these confidentiality alternatives may be overridden by individual voters, for example, whether an end user's vote is allowed to remain anonymous even if the general setting is "fully identified."

As will be understood by those familiar with the art, the present invention may be embodied in other specific forms without departing from the spirit or essential characteristics thereof. For example, a single computer could handle the tasks of vote processor, vote administrator, and voting unit. Accordingly, disclosures of the preferred embodiment of the invention is intended to be illustrative, but not limiting, of the scope of the invention which is set forth in the following claims.

What is claimed is:

1. A voting apparatus using a computer system for processing and reporting votes of voters, said voting apparatus comprising:

means for inputting a proposal and a set of terms into said computer system, wherein said inputting means are electrically coupled to said computer system, wherein said proposal requires voting by a group of voters;

a plurality of voting units electrically coupled to said computer system for inputting said votes, each of said voting units comprising a switching means for transmitting voter input signals to said computer system, wherein said voter input signals are selected from a group comprising a first form and a second form, said first form being a conditional input and said second form being an unconditional input;

means for processing each of said voter input signals to determine corresponding computed value signals, said computed value signals selected from a group consisting of:

a first computed signal having a unique value;

a second computed signal having multiple values; and

a third computed signal representing no solution;

means for applying said set of terms to the first, second and third computed signals to determine a set of results, said set of results selected from a group comprising at least:

a total of said first computed signals of a first type and of said second computed signals of said first type, wherein said first type is an affirmative value;

a total of said first computed signals of a second type and of said second computed signals of said second type, wherein said second type is a negative value;

a total of said first computed signals of a third type and second computed signals of said third type, wherein said third type is an abstention value;

a first solution comprising all values in a first set having the largest total of the first computed signals of said first type and of the second computed signals of said first type;

a second solution comprising all values in a second set having the largest total of the first computed signals of said second type and of the second computed signals of said second type;

a first series of solutions comprising a plurality of first ordered sets, said first ordered sets being arranged according to the sum of the number of first and second computed signals of said first type;

a second series of solutions comprising a plurality of second ordered sets, said second ordered sets being arranged according to the sum of the number of first and second computed signals of said second type;

a third series of solutions comprising a plurality of third ordered sets, said third ordered sets being arranged according to the sum of first and second computed signals of all of said types; and

a display means for presenting said results.

2. The voting apparatus of claim 1 wherein said set of terms comprises:

criteria for selecting a preferred solution from a group of possible solutions;

criteria for selecting a response when no solution is possible;

qualifications which must be met by a voter to be included in said group of voters;

deadlines for inputting voter input signals; and

allowed voter input signals.

3. The voting apparatus of claim 1 further comprising:

means for determining if said voter input signals meet said set of terms; and

means for rejecting voter input signals which do not meet said set of terms.

4. The voting apparatus of claim 1 wherein said plurality of voting units comprise:

a desktop voting unit;

a mobile voting unit;

a portable voting unit; and

a voting unit embedded in a separate computer.

5. The voting apparatus of claim 1 wherein said plurality of voting units are selected from a group comprising a first form and a second form, wherein said first form is a dedicated device and said second form is a general purpose device with multiple function capabilities.

6. The voting apparatus of claim 5 wherein said general purpose device comprises:

a personal computer; and

an interactive TV.

7. The voting apparatus of claim 1 wherein said first form and said second form of voter input signals are selected from a group comprising a first type and a second type, said first type being voter input signals that are equally weighted and said second type being voter input signals that are unequally weighted.

8. The voting apparatus of claim 1 wherein said voter input signals of said second form are selected from a group comprising affirmative, negative, and abstention.

9. The voting apparatus of claim 1 wherein said voter input signals of said first form are selected from a group comprising:

voter input signals dependent on said group of voters;

voter input signals dependent on a subset of said group of voters; and

voter input signals dependent on an event.

10. The voting apparatus of claim 1 wherein each of said voting units further comprises:

means for receiving output from the computer system, wherein said output includes the proposal, the set of terms, and the set of results; and

a display for presenting the output received from the computer system.

11. The voting apparatus of claim 1 wherein said voter input signals further comprise comments and modifications to said proposal.

12. The voting apparatus of claim 1 further comprising controller means, coupled to said computer system and said plurality of voting units, for accepting or rejecting additional voter input signals from an individual voter of said group of voters after an initial voter input signal has been provided by said individual voter, wherein said acceptance or rejection of said additional voter input signals is governed by said set of terms.

13. The voting apparatus of claim 12 wherein said additional voter input signals are provided by said individual voter before said processing of said voter input signals has taken place.

14. The voting apparatus of claim 12 wherein said additional voter input signals are provided by said individual voter after said processing of said voter input signals has taken place.

15. The voting apparatus of claim 12, further comprising means coupled to said controller means, for:

processing additional voter input signals;

determining a new set of results; and

displaying the new set of results.

16. The voting apparatus of claim 1 wherein said voter input signals of said first form include consensus building voter input signals.

17. The voting apparatus of claim 16 further comprising:

means for constructing a consensus building chart; and

means for displaying said consensus building chart.

18. The voting apparatus of claim 17 further comprising means of associating an individual voter of said group of voters with the voter input signal of said individual voter.

19. The voting apparatus of claim 1 further comprising:

means for comparing the set of results to a prescribed objective to determine a best solution; and

means for displaying the best solution.

20. The voting apparatus of claim 1 further comprising:

means for determining which of said voter input signals can be modified to eliminate computed indeterminate signals;

means for determining which of said voter input signals can be modified to eliminate computed unresolvable signals;

means for determining the form of modification required to eliminate computed indeterminate signals;

means for determining the form of modification required to eliminate computed unresolvable signals;

means for recommending said modifications to an individual voter of said group of voters; and

means for recommending said modifications to said group of voters.

21. The voting apparatus of claim 1 further including:

means for determining which voter input signals were most critical in obtaining a prescribed set of results;

means for displaying said most critical voter input signals;

means for determining which voter input signals were least critical in obtaining a prescribed set of results; and

means for displaying said least critical voter input signals.

22. The voting apparatus of claim 1 further comprising means for allowing an individual voter of said group of voters to prevent other voters of said group of voters from inputting voter input signals dependent upon the voter input signal of said individual voter.

23. A method of voting using a computer system, the method comprising the steps of:

inputting a proposal and a set of terms into the computer system;

displaying the proposal and the set of terms on a plurality of voting units;

inputting voter input signals into said voting units, wherein said voter input signals are selected from a group comprising a first form and a second form, said first form being a conditional input and said second form being an unconditional input;

transmitting the voter input signals from each voting unit to the computer system through an electrical network;

processing each of said voter input signals to determine corresponding computed value signals, said computed value signals selected from a group consisting of:

a first computed signal having a unique value;

a second computed signal having multiple values; and

a third computed signal representing no solution;

applying said set of terms to the first, second and third computed signals to determine a set of results, said set of results selected from a group comprising at least:

a total of said first computed signals of a first type and of said second computed signals of said first type, wherein said first type is an affirmative value;

a total of said first computed signals of a second type and of said second computed signals of said second type, wherein said second type is a negative value;

a total of said first computed signals of a third type and second computed signals of said third type, wherein said third type is an abstention value;

a first solution comprising all values in a first set having the largest total of the first computed signals of said first type and of the second computed signals of said first type;

a second solution comprising all values in a second set having the largest total of the first computed signals of said second type and of the second computed signals of said second type;

a first series of solutions comprising a plurality of first ordered sets, said first ordered sets being arranged according to the sum of the number of first and second computed signals of said first type;

a second series of solutions comprising a plurality of second ordered sets, said second ordered sets being arranged according to the sum of the number of first and second computed signals of said second type;

a third series of solutions comprising a plurality of third ordered sets, said third ordered sets being arranged according to the sum of first and second computed signals of all of said types; and

presenting said results.

24. The method according to claim 23 further comprising the steps of:
determining if said voter input signals meet said set of terms; and
rejecting said voter input signals that do not meet said set of terms.

25. The method according to claim 23 wherein said voter input signals of said second form are selected from a group comprising affirmative, negative, and abstention.

26. The method according to claim 23 wherein said first form and said second form of voter input signals are selected from a group comprising a first type and a second type, said first type being voter input signals that are equally weighted and said second type being voter input signals that are unequally weighted.

27. The method according to claim 23 wherein said voter input signals of said first form are selected from a group comprising:
voter input signals dependent on said group of voters;
voter input signals dependent on a subset of said group of voters; and
voter input signals dependent on an event.

28. The method according to claim 23 wherein said voter input signals further comprise comments and modifications to said proposal.

29. The method according to claim 23 wherein said set of terms comprises:
criteria for selecting a preferred solution from a group of possible solutions;
criteria for selecting a response when no solution is possible;
qualifications which must be met by a voter to be included in said group of voters;
deadlines for inputting voter input signals; and
allowed voter input signals.

30. The method according to claim 23 further comprising the step of accepting or rejecting additional voter input signals from an individual voter of said group of voters after an initial voter input signal has been provided by said individual voter, wherein said acceptance or rejection of said additional voter input signals is governed by said set of terms.

31. The method according to claim 30 wherein said additional voter input signals are provided by said individual voter before said processing of said voter input signals has taken place.

32. The voting apparatus of claim 30 wherein said additional voter input signals are provided by said individual voter after said processing of said voter input signals has taken place.

33. The method according to claim 30 further comprising the steps of:
processing additional voter input signals;
determining a new set of results; and
displaying the new set of results.

34. The method according to claim 23 wherein said voter input signals of said first form include consensus building voter input signals.

35. The method according to claim 34 further comprising the steps of:
constructing a consensus building chart; and
displaying said consensus building chart.

36. The method according to claim 35 further comprising the step of associating an individual voter of said group of voters with the voter input signal of said individual voter.

37. The method according to claim 23 further comprising the steps of:
comparing the set of results to a prescribed objective to determine a best solution; and
displaying the best solution.

38. The method according to claim 23 further comprising the steps of:
determining which of said voter input signals can be modified to eliminate computed indeterminate signals;
determining which of said voter input signals can be modified to eliminate computed unresolvable signals;
determining the form of modification required to eliminate computed indeterminate signals;
determining the form of modification required to eliminate computed unresolvable signals;
recommending said modifications to an individual voter of said group of voters; and
recommending said modifications to said group of voters.

39. The method according to claim 23 further comprising the steps of:
determining which voter input signals were most critical in obtaining a prescribed set of results;
displaying said most critical voter input signals;
determining which voter input signals were least critical in obtaining a prescribed set of results; and
displaying said least critical voter input signals.

* * * * *

APPENDIX H

USPN 4,210,962

United States Patent [19]

Marsh et al.

[11] **4,210,962**

[45] **Jul. 1, 1980**

[54] **PROCESSOR FOR DYNAMIC PROGRAMMING**

[75] Inventors: **James P. Marsh**, San Jose; **Howard L. Steadman**, Santa Cruz, both of Calif.

[73] Assignee: **Systems Control, Inc.**, Palo Alto, Calif.

[21] Appl. No.: **920,970**

[22] Filed: **Jun. 30, 1978**

[51] **Int. Cl.**[2] **G06F 15/46**

[52] **U.S. Cl.** **364/402**; 364/105; 364/300; 364/900

[58] **Field of Search** 364/105, 200, 300, 402, 364/900

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,411,140	11/1968	Halina et al.	364/900
3,974,481	8/1976	Ledieu et al.	364/402 X

OTHER PUBLICATIONS

R. E. Larson et al., "Parallel Processing Algorithms for the Optimal Control of Non-Linear Dynamic Systems," *IEEE Trans. on Computers*, vol. C-22, No. 8, Aug. 1973, pp. 777–786.

P. A. Gilmore, "Structuring of Parallel Algorithms," *Journal of the Association for Computing Machinery*, vol. 15, No. 2, Apr. 1968, pp. 176–184.

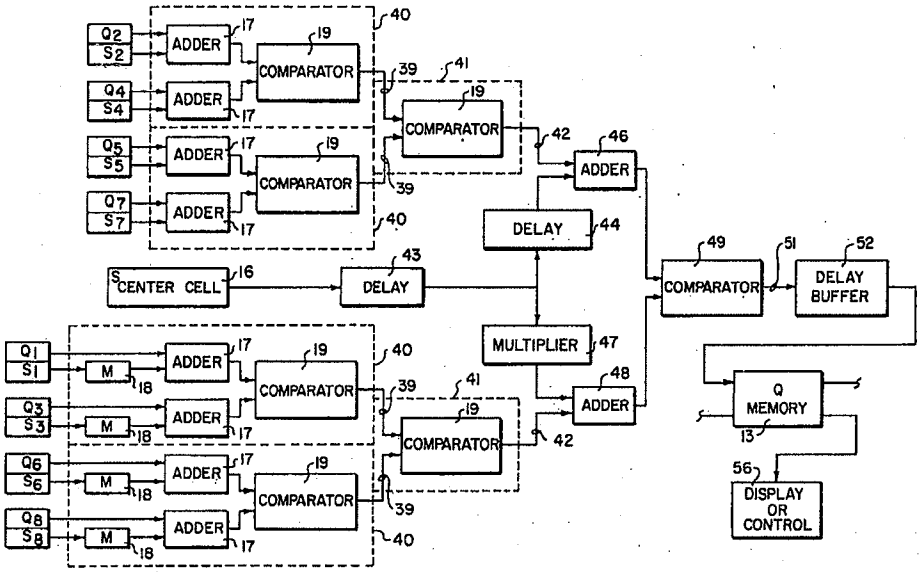
Primary Examiner—Jerry Smith

Attorney, Agent, or Firm—C. Michael Zimmerman

[57] **ABSTRACT**

A parallel/pipeline processor is described designed to quite rapidly solve optimization problems with dynamic programming. The state variables and optimum costs associated with a transition are presented simultaneously to a plurality of parallel comparators at the base of a tree of comparators. Such values are presented in a serpentine memory arrangement which sequentially advances the values between the base comparators to enable the comparator tree to compute optimum transitions for successive state space cells in a pipeline fashion.

10 Claims, 8 Drawing Figures



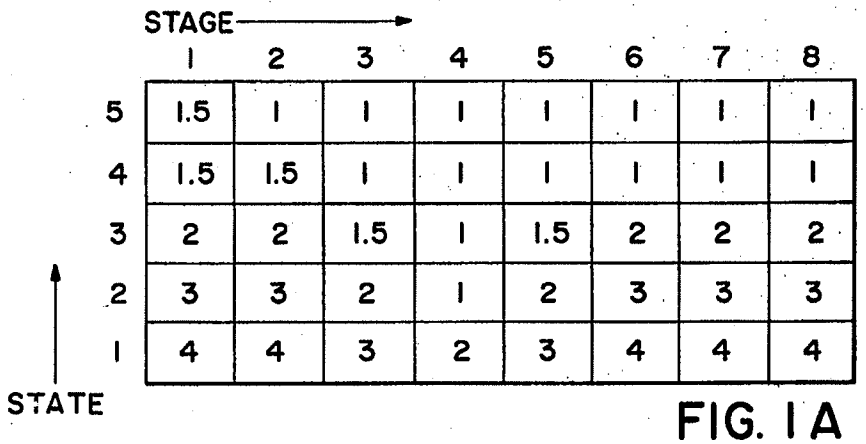


FIG. 1A

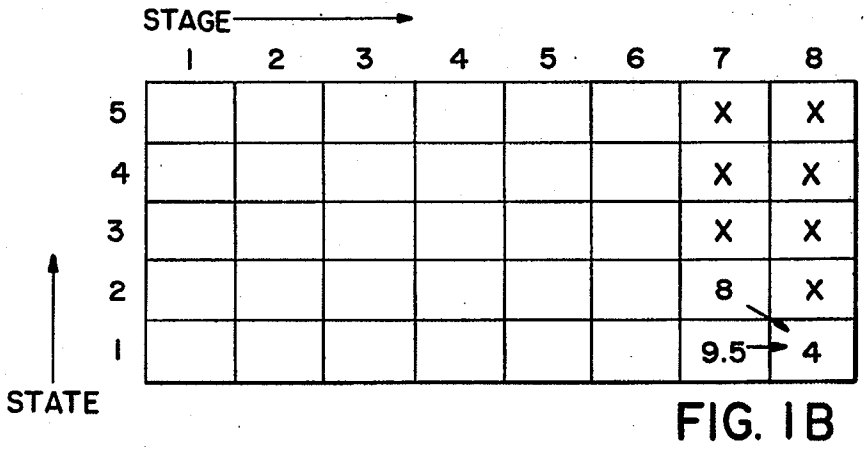


FIG. 1B

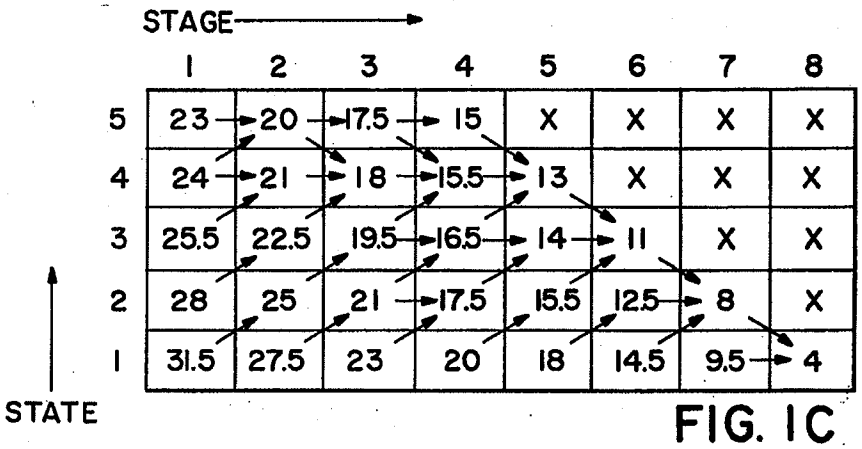


FIG. 1C

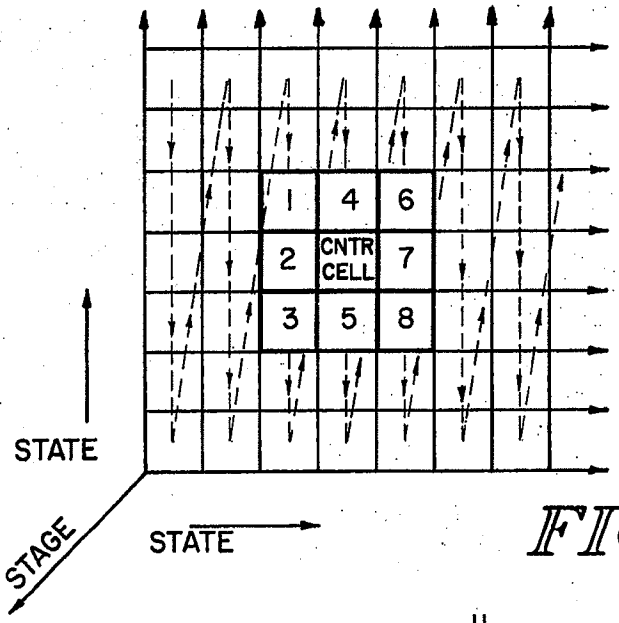

1
4
6
2
CNTR CELL
7
3
5
8
STATE
STAGE
STATE

FIG. 2

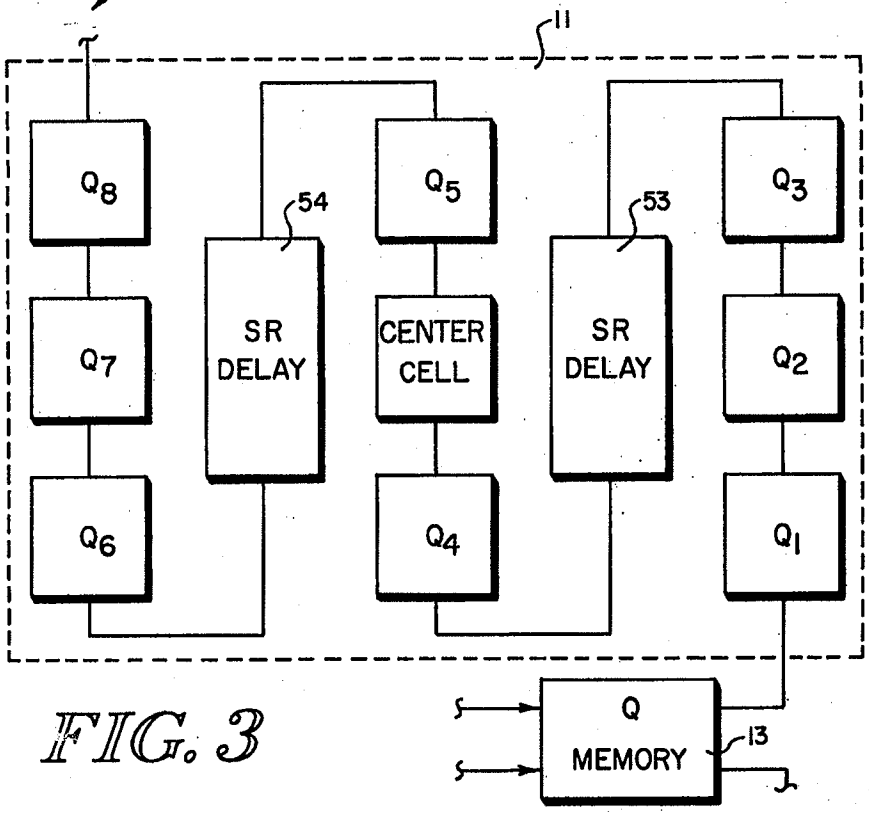

11
Q8
Q7
Q6
54
SR DELAY
Q5
CENTER CELL
Q4
53
SR DELAY
Q3
Q2
Q1
Q MEMORY
13

FIG. 3

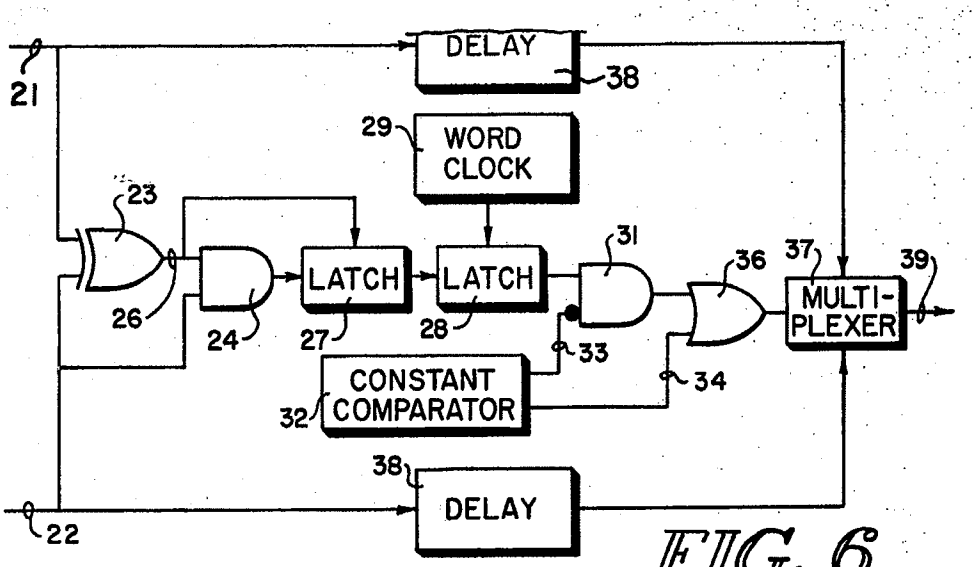

DELAY
38
21
29
WORD CLOCK
23
31
36
37
39
LATCH
LATCH
MULTI-PLEXER
26
24
27
28
33
34
CONSTANT COMPARATOR
32
38
DELAY
22

FIG. 6

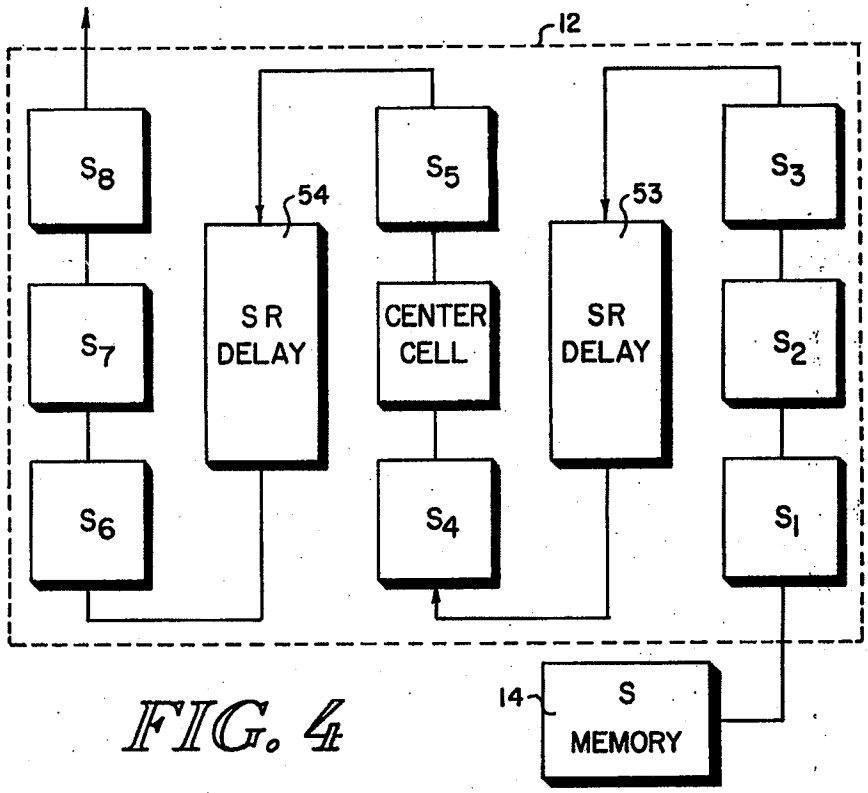
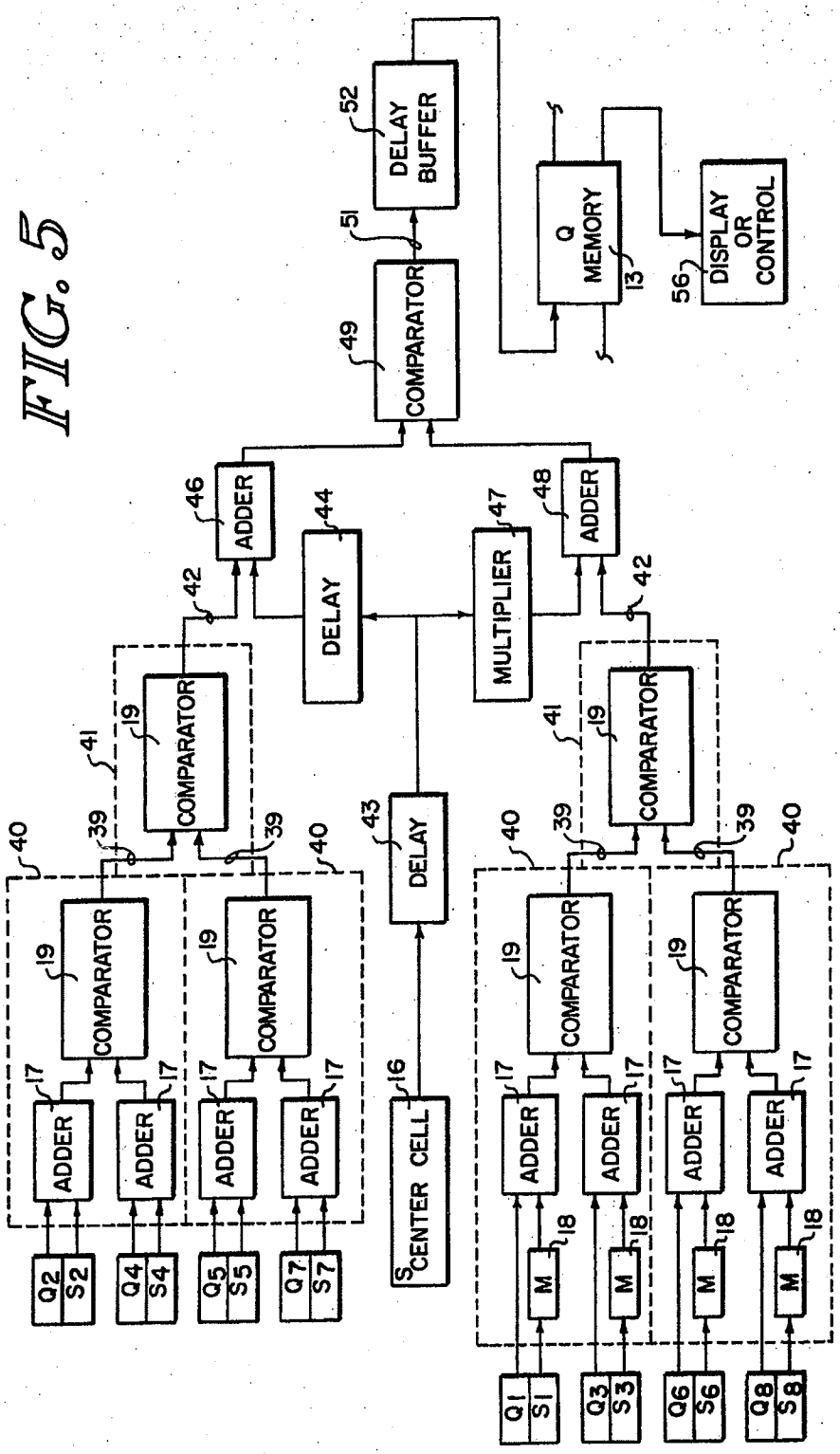

12
S8
S7
S6
54
SR DELAY
S5
CENTER CELL
S4
53
SR DELAY
S3
S2
S1
14
S MEMORY

FIG. 4

FIG. 5


52
DELAY BUFFER
51
49
COMPARATOR
Q MEMORY
13
56
DISPLAY OR CONTROL
46
ADDER
44
DELAY
47
MULTIPLIER
48
ADDER
42
42
41
19
COMPARATOR
41
19
COMPARATOR
40
39
39
40
43
DELAY
40
39
39
40
19
COMPARATOR
19
COMPARATOR
19
COMPARATOR
19
COMPARATOR
17
ADDER
17
ADDER
17
ADDER
17
ADDER
16
CENTER CELL
17
ADDER
17
ADDER
17
ADDER
17
ADDER
18
M
18
M
18
M
18
M
Q2 S2
Q4 S4
Q5 S5
Q7 S7
S
Q1 S1
Q3 S3
Q6 S6
Q8 S8

PROCESSOR FOR DYNAMIC PROGRAMMING

BACKGROUND OF THE INVENTION

The present invention relates to processors and, more particularly, to a processor having a parallel pipeline architecture designed especially to solve dynamic programming problems rapidly.

Dynamic programming provides a sequential decision-making approach for solving optimization problems. For example, the minimum fuel path of an aircraft between two locations often can be determined by using dynamic programming. In basic terms, dynamic programming converts a comparison of the many overall choices to select the optimum choice into a sequential comparison of much simpler intermediate optimization problems. In the aircraft example, the space (typically called a "state space") through which the aircraft can travel is divided into a multiple number of space volumes or cells. The optimum transitions from one cell to each of the adjacent ones closer to the end point is then calculated. The optimum transition may be dependent, for example, on the altitude and velocity of the aircraft.

The candidate optimum transitions of the aircraft from each and every one of the cells within which it can exist in the state space between the starting and end points are compared, and these optimum transitions are utilized to calculate the overall optimum path between the two points. Dynamic programming typically is carried out in reverse, i.e., by starting the calculations at the cell containing the end point and performing the calculations in reverse therefrom.

While the above example is based on the path of an aircraft between two points, dynamic programming is applicable to many other types of optimization problems in which a state space (not necessarily representative of physical space or volume) can be divided into cells. Manufacturing process control problems which can be characterized by a sequential solution of simpler intermediate optimization problems are also subject to dynamic programming solution. More details and specific explanations of dynamic programming can be found in the books entitled *Applied Dynamic Programming* by R. E. Bellman and S. E. Dreyfus, Princeton University Press (1962) and *State Increment Dynamic Programming by R. E. Larson, American Elsevier Publishing Company, Inc., (*1968) New York, N.Y.

While dynamic programming is a powerful tool for solving optimization problems, it has not found any significant usage in real-time applications in which solutions are required or desired relatively rapidly. For example, before the present invention, dynamic programming could not, as a practical matter, be used to determine the optimum path for a moving aircraft in real-time. The difficulty is that conventional digital computers on which such programs are solved are limited to serial solutions. It will be recognized that to solve a dynamic programming problem, it is necessary to serially make numerous computations, each one of which is dependent upon the results obtained in earlier comparisons. In any real-time situation, even with very fast processing speeds, it often takes more than a minute to obtain a solution. Again considering an aircraft, if an optimum path to some location from an aircraft's present position was chosen by a dynamic programming solution using conventional techniques, by the time the solution was obtained, the aircraft would have so materially changed its position the solution would be relatively meaningless.

Although parallel algorithms for dynamic programming problems have been described in the past (see the paper entitled "Parallel Processing Algorithms for the Optimal Control of Non-Linear Dynamic Systems" by R. E. Larson and E. Tse, *I.E.E.E. Transactions on Computers,* Volume C-22, No. 8, August 1973, pages 777–786), sufficiently adequate methods of organizing the data and passing it between computational elements have not been provided prior to the instant invention to permit the design of a parallel dynamic programming device. That is, the intermediate solutions required for optimum transitions from adjacent cells often requires the same information. The resulting necessity for interprocessor communication in any conventional arrangement has added more complexity and time to dynamic programming device designs to warrant their use.

SUMMARY OF THE INVENTION

The present invention provides a processor especially designed to solve dynamic programs in a minimum of time and without the necessity of complex interprocessor communication. It accomplishes this by providing separate computation means to simultaneously make the comparisons necessary to determine the optimum one of the numerous transitions which can be made to or from one cell to any allowable one adjacent thereto. In this connection, it includes a memory arrangement which simultaneously presents to a plurality of computation means, the data associated with various pairs of candidate transitions to or from a cell being investigated. As a particularly salient feature of the instant invention, the computation means are so related to one another, and the memory means is so designed, that computations for the candidate transitions to or from cells which are adjacent to one another can be made simultaneously without contention for data by the different computation means. In a very basic sense, the processor of the invention can be referred to as a parallel/pipeline computer architecture.

The invention includes other features and advantages which will be described or will become apparent from the following more detailed description of a preferred embodiment.

BRIEF SUMMARY OF THE DRAWING

With reference to the accompanying four sheets of drawing:

FIGS. 1(*a*)–(*c*) provide a simple example of dynamic programming to facilitate an understanding of the present invention;

FIG. 2 is a representation of a state space and the manner in which it is scanned by a preferred embodiment of the invention;

FIGS. 3 and 4 are block diagrams illustrating a memory arrangement for presenting data to the computation architecture of FIG. 5;

FIG. 5 is a block diagram of a preferred embodiment of the computer architecture of the invention, designed to solve a two-state, aircraft path dynamic program; and

FIG. 6 is a schematic logic diagram illustrating in more detail the design of a comparator circuit for the initial comparison of a pair of transitions from a cell in a state space.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

For an understanding of the present invention and a true appreciation of the advance in the state of the art it represents, a fundamental understanding of dynamic programming is believed to be necessary. As mentioned previously, dynamic programming is an approach for solving optimization (maximization or minimization) problems, relying on dissecting the main optimization problem into many intermediate optimization problems. In other words, dynamic programming converts the simultaneous determinations of an entire optimization into a plurality of sequential solutions of vastly simpler intermediate optimization problems. The resulting overall solution is precisely the same solution which would be obtained if one exhaustively checked all possible transition combinations, and is obtained without actually performing such a computationally prohibitive search. A dynamic programming problem is characterized by the fact that the decision made at one time affects the decisions made later in the process, and is not just a decision unto its own.

In explicit dynamic programming, the overall problem is divided into "stages" of a "state space". (The term "dimension" or "dimensions" are sometimes used as an alternative to the term "state space". However, neither dynamic programming nor the instant invention is in any way limited to problems in which the state space represents geometric dimensions, and for this reason the term "state space" will be used herein to avoid possible confusion.) The state space is itself divided into a plurality of adjacent "cells". Implicit dynamic programming is essentially the same as explicit dynamic programming, except that the stages of a state space are replaced, in effect, by additional state space variables. The instant invention is described in connection with explicit dynamic programming.

Generally, in a dynamic programming problem the goal is to advance from a state space cell in one stage to a state space cell in another stage until the cell in the stage and state space representative of the final solution is reached. The issue with respect to each transition from one stage to another is into which state space cell of the new stage should the transition be made in order to achieve the optimum transition. It should be noted that while the problem described herein solved by the instant invention requires an advancement from one cell in the state space to an adjacent cell at the next stage, from the conceptual standpoint dynamic programming is not so limited. That is, some problems may permit transition to any cell in the next stage. The instant invention is equally applicable to the solution of such other problems.

The basic features characteristic of most problems suitable for solution by dynamic programming are as follows:

1. the overall problem can be divided into stages, with a policy decision required at each stage;

2. each stage has a number of states (state space cells) associated therewith;

3. the effect of the policy decision at each stage is to transform the current state into a state associated with the next stage (possibly according to a probability distribution);

4. given the current state, the optimum policy decisions for the remaining stages is independent of the policy adopted in previous stages;

5. the solution procedure begins by finding the optimum policy for each state at either the beginning or last stage;

6. a recursive relationship is available which identifies the optimum policy for each state space with a specified number of stages remaining, given the optimum policy for each state with the specified number of stages remaining, less one; and

7. using this recursive relationship, the solution procedure moves from stage to stage until the optimum overall decision is made.

The equations of any dynamic system undergoing dynamic programming can be linear or non-linear and time-varying. The criteria on which the optimization is based can be quite arbitrary. Constraints of a wide variety are permitted. Random effects of a very general nature can be rigorously taken into account, by using the same analytic formulation. In addition, the variables representing a state space are almost always chosen to provide an absolute optimum in each state space, rather than a local optimum. Moreover, most dynamic program solutions specify the optimum transition at each state of the system for every stage; thus, a dynamic programming solution can be implemented as a feedback (closed-loop) controller, in which the state of the system is constantly measured and the corresponding optimum control is applied.

FIGS. 1(*a*) through 1(*c*) are included to illustrate graphically dynamic programming applied to a simple, one-state optimization problem. Let it be assumed that we are interested in knowing the flight path of an aircraft to a specified landing field location which will use the minimum amount of fuel. The aircraft of concern is capable of changing altitude but is incapable of turning either left or right, i.e., the aircraft will start out and stay in a single geometric plane containing the starting location and the landing field. Let it further be assumed that the "state space" in which the aircraft is to end its flight is known, but there are a plurality of different airfields or "cells" from which the aircraft could start. Thus, "reverse" dynamic programming is a more suitable tool for solving the optimization problem, than "forward" dynamic programming.

The grid pattern illustrated in FIGS. 1(*a*) through 1(*c*) represents a typical state space for such an aircraft problem. As illustrated, there are eight stages, each one of which is made up of five state cells. In this particular problem, the stages correspond to intervals along the ground track, and the states correspond to the altitude of the aircraft. The "cost" of flying in any particular state space cell is proportional to the amount of fuel needed to fly at the altitude represented thereby. The numbers inserted in each of the state space cells of FIG. 1(*a*) represents such cost.

Transitions of the aircraft are limited to movement thereof from the cell in the stage in which it finds itself to a cell of the succeeding stage adjacent to the cell within which it is in. As will be recognized, this will require the aircraft to either gain altitude diagonally, remain at the same altitude, or lose altitude diagonally. Thus, there is also a cost in fuel required for the transition itself. Let us assume that each of these transition costs can be represented by the unit **1** for a loss of altitude, the unit 1.5 for staying at the same altitude, and the unit 2.5 for a gain of altitude. The total cost for a transition to a new cell in an adjacent stage is the sum of the cost of flying in such cell and the cost of making the

transition thereto. This total cost will be identified herein by the letter "S".

While the total cost in this particular problem includes an actual physical transition of the aircraft from one state space cell to another, the term "transition" as used herein is meant not only to encompass data associated with such an actual transition but also data defining one or more other variables associated with a state space cell being checked.

Let it be assumed that the landing field is in the first state cell of the last stage, i.e., stage 8. Thus only state 1 in such stage is "allowable". And, the only states in stage 7 which are allowable are those two states adjacent to state 1 in stage 8, i.e., states 1 and 2 of stage 7.

The first step is to calculate for each of the allowable states (or cells) at stage 7, the cost of a transition to state 1 of stage 8. FIG. 1(*b*) provides the results. The numerals in the first two cells of stage 7 indicate the total cost in fuel associated with a transition through such states and a transition to the end, stage 8, state 1. The arrows represent the optimum transition paths. Since there is only one allowable path from each of the two states in stage 7 to the end state in stage 8, the arrows in this particular situation represent the only path. The numerals "8" and "9.5" in the two states 1 and 2 (stage 7) represent, respectively, the total overall cost associated with the transition from the cells within which they are found, to the last stage. Notice that this overall cost to the last state in the last stage has replaced the cost set forth in FIG. 1(*a*) associated with each cell. This overall cost to the end is referred to herein as "Q".

Continue calculation for each stage in the reverse direction, at each stage calculating, for every allowable state, the best state in which to go in the succeeding stage. Note that this single stage optimization is "globally optimum" in the sense that the best path and total cost for each state from the next stage to the end has already been calculated.

FIG. 1(*c*) presents the total cost to the end ("Q") for all allowable states at all stages. Note that there are transition ties, i.e., transitions from one stage to the next succeeding state can be optimally made to more than one state space cell. Such transitions are indicated by more than one arrow extending from a cell in one stage into cells of the next succeeding stage. In actual practice, round-off errors will typically cause two equal paths to be slightly different, resolving such potential conflicts.

The overall optimum path is indicated by the minimum total cost set forth in a cell of stage 1. This minimum is "23" in state 5 of stage 1, and the optimum path is indicated by the arrows extending from such state to state 1 of stage 8.

It should be noted that although in the above example the various costs were added to arrive at an actual cost, and thus the "cost function" is the sum of the cost at each stage, in some problems, the cost function may be the product of the costs at each stage. Such calculations are often simplified in dynamic programming by employing as costs the logarithms of the actual values involved, so that addition can be substituted for the multiplication. There are, however, other functions on the cost besides addition and multiplication, which could be optimized, and the instant invention is not function limited.

Whereas the above simple example of dynamic programming is based on a one-state (or one dimension) problem, most practical problems require dynamic programming of a plurality of state dimensions. The preferred embodiment of the invention will be described with an architecture for solving a two-state problem involving aircraft flight. However, it will be recognized by those skilled in the art that the principles on which the architecture is based are applicable to solving optimization problems with dynamic programming utilizing any number of states. The architecture need only be expanded to handle additional states, or retracted to handle a single state.

FIG. 2 graphically illustrates a two-state aircraft optimization problem and how the architecture of the instant invention is related thereto. This two-state problem is like the aircraft problem used earlier, except that the aircraft can fly either to the left or right, i.e., state space cells in adjacent planes are allowable. In this connection, note that the stage axis extends perpendicular to the page of the drawing, whereas the two state axes are respectively horizontal and vertical lines in such plane. The sequential stages can be thought of as sequential planes parallel to the plane containing the sheet of the drawing, with succeeding stages in the direction of aircraft travel being successively farther away from the viewer. To determine the optimum transition from any specified state space cell at a given stage to the next stage using reverse dynamic programming, transitions to all adjacent cells in the next stage (in front of the drawing sheet) must be checked. This is with the assumption that the problem is one in which only transitions to adjacent cells are allowed.

Turning to FIG. 2, it is a checking of the cells surrounding the center state space cell that will be used to describe the instant invention. Only transitions from the center cell to such cells in the stage illustrated, referred to by the numerals 1 through 8, will be allowed. Transitions from the center cell of the stage illustrated to the center cell being checked of the next succeeding stage are considered trivial and are not allowed in the problem for which the preferred embodiment of the invention was designed.

Each of the allowable state space cells in the illustrated stage has at least two variables associated therewith, a "Q" variable representing the optimum transition to the end of the problem and an "S" variable representing the cost associated with the state space cell itself. It is to be remembered that such cost variable "S" may itself be a function of a plurality of cost variables, such as the cost of moving into the state and the cost of moving through the same, as in the example described previously.

The present invention enables the transitions from the center cell in the stage illustrated, to the surrounding cells being checked in the next stage, to be determined simultaneously. To this end, the invention includes means to present simultaneously data associated with transitions from the state cell being checked to each of the allowable state cells illustrated in FIG. 2. That is, the serpentine memories 11 and 12 as represented in FIGS. 3 and 4 by the dotted line boxes present respectively the "Q" and "S" values for the allowable state space cells in a serial fashion. These values are obtained from main "Q" and "S" memories 13 and 14, respectively, which memories retain all of the calculated Q's and given S's. Although these memories are represented in the drawing by two separate blocks, it will be recognized that most probably they would be provided as separate space allocations in a single memory.

Each serial memory word location in the serpentine memories of FIGS. 3 and 4 is tied to the two-state representation of FIG. 2. For example, the box labeled "Q_8" in the serpentine memory of FIG. 3 represents the location of the Q value for the state space indicated by "8" in FIG. 2. Similarly, the location in serpentine memory 12 of the word representing the "S" value for state space 8 in FIG. 2 is represented by the box "S_8" in FIG. 4. The other word locations in the two memories are likewise labeled. Each of the serpentine memories 11 and 12 also include shift registers at appropriate locations as will be discussed, to separate the data flowing through the serpentine memories in connection with a particularly salient feature of the instant invention to be described.

FIG. 5 schematically illustrates the computational architecture of the present invention. The word locations in the serpentine memories for each of the "Q" and "S" values associated with the allowable state spaces surrounding the cell being checked is represented at the left in the figure. It will be noted that all of the word locations illustrated above the box 16 labeled "center cell" are of cells from which diagonal transitions to the state space cell being checked are required, whereas all of the word locations represented below the box 16 are those for cells requiring horizontal or vertical transitions to the cell being checked. The purpose for this segregation will be described hereinafter.

The "Q" and "S" values of each allowable state space cell adjacent the state space cell being checked are combined to provide a total optimum for such cell. This addition is represented in FIG. 5 by adder blocks 17.

In the particular dynamic problem for which this preferred embodiment was designed, the time required to traverse each state space cell diagonally is greater than the time required to traverse each cell horizontally or vertically; thus there is a corresponding additional fuel cost associated with a diagonal transition. Such additional cost must be taken into account in computing the optimum transition. It is taken into account, insofar as the cells of the stage illustrated in FIG. 2 is concerned, by multiplying the logarithm of the "S" value of each of the state space cells which would require a diagonal transition from the cell being checked, by the ratio of the diagonal transit time to the direct transit time. This multiplication is represented in FIG. 5 by the blocks 18 positioned between the outputs of each of the serpentine memory locations representing diagonal transitions, and its associated adder 17.

The respective outputs of each pair of adders 17 is compared to determine which of the two transitions represented thereby is the optimum one. In this connection, such outputs are connected to the inputs of an associated comparator 19. The compare decisions made at this time should take into account whether or not the state cell being checked is at an edge of the state space, and whether or not each of the adjacent cells of the next stage are allowable. FIG. 6 illustrates suitable comparator circuitry for each of the comparators 19. The words of data emanating from the associated adders 17 respectively appear on the lines 21 and 22. Each of such words are made up of a plurality of bits defining not only the absolute value of the combined term, but also the optimum transition direction for the "Q" value of the cell represented thereby. The bits defining these words are presented serially on the lines 21 and 22, and an exclusive OR gate 23 samples corresponding bits on the lines 21 and 22 defining the absolute values of the combined terms. The logic made up of exclusive OR gate 23 and AND gate 24 will indicate any discrepancy between such bits. That is, the output of the exclusive OR gate 23 appearing on line 26 will have one voltage state when the bits are the same, and another state when such bits are different. This output is compared by AND gate 24 to the bit appearing on line 22, for example, to determine which line should be selected. If there is a discrepancy, the output of AND gate 24 will be stored in a latch 27, which latch is strobed by the output of the exclusive OR gate 23.

It is the last bit discrepancy which indicates finally which of the words appearing on lines 21 and 22 is to be selected. This is because the last bit of each word is the most significant bit. The state indicated by latch 27 is fed to and stored within a latch 28, upon a word clock 29 indicating that the ends of the words appearing, respectively, on lines 21 and 22 have been reached.

"Override" structure is included in the comparator 19 to assure that improper optimums are not obtained either because a transition to one of the cells is not allowable in the problem or because the cell being checked is adjacent an edge of the state space. More particularly, the state of latch 28 is fed to one input of an AND gate 31. The other input of the AND gate 31 is controlled by a memory index for the cell being checked. This is represented in the drawing by constant comparator box 32. This box will place an output on either of its two output lines 33 and 34 depending upon whether or not the cell being checked is at an edge space. Its output on line 33 will normally allow the state stored in latch 28 to be passed through the AND gate 31 to an OR gate 36. However, when the cell being checked is adjacent to an edge of the cell space, or one of the cells represented by the words on line 21 and 22 is not allowable, the logic formed by the combination of the AND gate 31 and the OR gate 36 will force the comparator to choose only the allowable cell state. In this connection, the output of gate 36 is fed to the control input of a multiplexer 37. The words appearing serially on the lines 21 and 22 are fed, through delays 38 designed to take into account the time delay involved in the selection process, to opposite data input terminals of the multiplexer 37. The word representing the optimum will be the only word which will be passed by the multiplexer 37 to its output as represented at 39 (FIGS. 5 and 6).

It will be recognized from the above that each pair of adders 17 (and their attendant multipliers where applicable) along with their associated comparator 19, acts, in effect, as computation means to compare data representative of transitions to or from each one of a pair of states to determine the optimum one of such pair. Such computation means are enclosed individually in dotted line boxes indicated by the reference numeral 40.

It should be noted that while in this embodiment the data which defines the direction associated with the Q value of each cell accompanies the absolute value of the Q through the various computations, it can instead, if desired, be generated by the various computations required to select the optimum transition.

The output of each pair of computations means 40 is fed to further computation means enclosed by the dotted line box 41. Each computation means 41 includes a comparator 19 (the same as the earlier described comparators 19), and selects which of the transitions represented by its two inputs is the optimum one, and passes the data representing the same to its output on line 42.

It will be noted that the outputs on the two lines **42** will represent the respective direct and diagonal optimum transitions. As mentioned previously, there is an additional transition cost associated with a diagonal transition. While this additional cost was taken into account insofar as the stage from which a transition to the cell being checked is to occur, the additional cost for traversing the cell being checked has not yet been taken into account. It will be noted, though, that this additional cost will be the same for all of the diagonal transactions and need not be taken into account until just before the optimum diagonal transition is compared with the optimum direct transition. To this end, the preferred embodiment being described includes means to adjust the value of such optimum transitions. More particularly, the "S" value of the cell being checked, the center cell **16**, is added or multiplied by a constant and then added, as appropriate, to the direct and diagonal optimum transitions prior to the same being compared. The result is delayed, of course, as represented by the blocks **43** and **44**, to assure that the value will reach the adder **46** which combines the terms, at the same time as the output of the comparator **41** which ultimately selected the optimum direct transition.

The constant which is multiplied by the "S" value of the center cell is the ratio of the diagonal transit time to the direct transit time. This multiplication is represented in FIG. **5** by multiplier **47** positioned between delay **43** and the adder **48**.

The optimum direct and the optimum diagonal transitions are compared to one another for selection of the overall optimum transition. This is represented by the output of the adders **46** and **48** being fed to comparator **49**. Comparator **49** can include the same logic as the comparators **19**. The overall optimum transition will then appear on the output line **51** of comparator **49**. This overall optimum transition will be in the form of a new "Q" value for the cell being checked, as well as data representing the direction of the transition to the cell which must be taken to achieve the optimum transition, i.e., the data will also indicate from which cell in the stage illustrated in FIG. **2** the transition must be made in order to achieve the optimum "Q" value which is so represented.

It should be noted that there may be other constant factors which need not pass through the full comparator tree prior to being taken into account. For example, it may be in an aircraft problem that several different aircraft speeds could be used in making transitions. The optimum transition could be selected neglecting the speed factor, and then the speed factor taken into account to select the ultimate optimum transition.

The data defining the optimum transition for the cell being checked will be fed into the Q memory **13** to replace the earlier Q value for the cell. This rewriting of the data in the Q memory, though, has to be delayed until such time as the old Q value for the cell is no longer needed in the checking of those cells adjacent the center cell being checked. This delay is provided by a delay buffer **52**.

It will be appreciated by the above that the invention provides parallel processing in the sense that all of the transitions for the cell being checked are compared simultaneously. That is, all of the computation means **40** operate simultaneously on data supplied thereto via the serpentine memories. This data defines the transitions from the cell being checked to all of the cells of the next stage.

As a particularly salient feature of the instant invention, it not only provides parallel processing for the center cell being checked, but enables those cells adjacent to such center cell to also be checked contemporaneously therewith. In this connection, it will be noted that while all of the computation means **40** will provide their computations simultaneously, once they have completed their comparison cycles such computation means are no longer required to determine the optimum transition for the cell being checked. It is therefore comprehended within the instant invention that simultaneously with the comparisons by the computation means **41** of the results provided by the associated computation means **40**, the computation means **40** are used to compare transitions in the determination of the optimum transition for another cell. This simultaneous usage of the successive computation means for the determination of the optimum transitions. to adjacent cells is continued for the full length of the computation "tree".

The contemporaneous checking for the optimum transitions to a plurality of cells is made quite simple and efficient by the combination of the serpentine memories with the computation architecture discussed above. In this connection, reference is made to FIG. **2** which illustrates the manner in which the cells whose transitions to the state space illustrated preferably are successively checked, i.e., the manner in which the state space is scanned. This scanning is represented in the figure by the dotted line arrows and, as can be seen, is similar to raster scanning in the sense that such scanning occurs column by column with retrace lines between such columns. Each of the serpentine memories is arranged to coincide with such scanning. That is, as can be seen by referring to FIGS. **3** and **4**, the order of the address locations for the Q and S values coincides with the order in which the cells adjacent to the cell being checked are encountered during the scanning. Thus, the Q and S values for the cells labeled "**1**", "**2**" and "**3**" are at the ends of the respective serpentine memories. Each of the memories further includes a shift register delay **53** positioned between the address locations for the respective cells labeled "**3**" and labeled "**4**". For the particular state space illustrated in FIG. **2**, the shift register delay should be four cells long and contain the respective Q and S values for the four cells of the state space which are scanned between the time the cells labeled "**3**" and "**4**" are encountered. The address locations for the Q and the S values of the center cell are positioned in the serpentine memories between the locations for the corresponding values of the cells labeled "**4**" and "**5**". It should be noted that only the S value of the center cell is required in connection with the optimization computations, and it is only such value which is tapped by the computation arrangement illustrated in FIG. **5**.

Another shift register delay **54** is disposed between the address locations in the respective serpentine memories for the Q and S values for the cells labeled "**5**" and "**6**". The purpose of such shift register is the same as that of the shift register **53**, i.e., to provide the appropriate scanning space between the cells labeled "**5**" and "**6**".

The serpentine memories are strobed to sequentially advance the value of each of the locations into the next location upon completion of each comparison cycle. The result is that the apparatus is automatically adjusted to begin checking for the optimum transitions to the cell immediately adjacent (in the case of FIG. **2**, immedi-

ately below) the cell undergoing checking. Thus, as soon as the computation means **40** are finished with their comparisons, they are automatically readied to accept values to be compared for the next cell to be checked. It will therefore be seen that the combination of the serial serpentine memories with the computation architecture described provides in a simple manner an arrangement for not only providing the initial comparisons in parallel via the computation means **40**, but also determining the optimum transition of adjacent cells in a pipeline fashion. An optimum transition for a cell will emanate from the comparator **49** at each word clock and be fed to the delay buffer **52**. The optimum transitions are returned to the Q memory **13** as previously mentioned, for storing for later use. This later use, which typically takes the form of a display or control, is represented in FIG. **5** by box **56**.

It will be seen from the above that since the processor of the invention is a parallel/pipeline device, it is capable of solving dynamic programming problems much more rapidly than serial computers. Because of such capability, the processor is usable to control in near real-time, processes and other functions which can be optimized using dynamic programming. It is also usable to solve certain types of detection and tracking problems in near real-time which heretofore have been incapable of solution by dynamic programming. Depending upon the particular problem to be solved, solutions can be obtained with the instant invention as much as one hundred times as fast as one can be obtained using conventional serial computation techniques. And as the dynamic programming problem becomes more complex, i.e., includes more and more state variables, the speed-up become even more dramatic.

While the invention has been described in connection with a preferred embodiment, it will be appreciated by those skilled in the art that various changes and modifications can be made. In this connection, the preferred embodiment has been described and designed especially for a two-state variable problem. It will be recognized, however, that it is easily within the skill of the art to expand the architecture to accommodate problems having more than two state variables. Moreover, the various computation means could be in the form of programmable microprocessors or the like, especially in a general purpose parallel/pipeline processor. And while the preferred embodiment has been described in connection with explicit dynamic programming, the invention is equally applicable to solving implicit dynamic programming problems. In view of these changes and modifications, it is intended that the coverage afforded applicants be limited only by the claims and their equivalent language.

We claim:

1. In a processor for solving dynamic programming problems, the combination comprising:

A. first computation means to compare data representative of candidate transitions to or from each one of a first plurality of states of a stage of a dynamic programming problem, to determine the optimum one of said first plurality of state transitions;

B. second computation means to compare data representative of candidate transitions to or from each one of a second plurality of states of said stage of dynamic programming problem, to determine the optimum one of said second plurality of state transitions; and

C. third computation means to compare data representative of the optimum ones of said state transitions of said first and second pluralities, simultaneously with comparison by said first and second computation means of data respectively representative of transitions to or from states of third and fourth pluralities of states of a successive stage of said dynamic programming problem to determine the optimum one of said state transitions of said third and fourth pluralities.

2. A processor according to claim **1** wherein said data representative of transitions to or from each state is based on a plurality of variables, one of which is representative of transitions to or from at least one other state.

3. A processor according to claim **2** wherein each of said variables representative of a candidate transition to or from another state is representative of a candidate future transition to another state.

4. A processor according to claim **2** wherein said data representative of transitions to or from each of said states of said first and second pluralities of states is representative of a plurality of different variables associated with transitions to each of said states, with the data representative of each of said variables being the value of the logarithm of the actual variable value; and said first and second computation means each further includes an adder for combining logarithmic values of the variables associated with each transition prior to comparison of the same with similar data associated with a transition to or from another state.

5. A processor according to claim **1** further including means to present simultaneously to said first and second computation means said data respectively associated with said transitions to or from said first and second pluralities of states for simultaneous determination of said optimum transitions of said first and second pluralities.

6. A processor according to claim **5** wherein said means to present said data simultaneously to said first and second computation means comprises a serial memory storing said data in a serial fashion adapted to present simultaneously to said first and second computation means, data stored at a plurality of serially related locations therein.

7. A processor according to claim **6** wherein said serial memory is adapted to sequentially advance said data between said serially related locations as said first, second and third computation means makes said comparisons, whereby data representative of different transitions is presented for comparison to said first and second computation means upon completion by said first and second computation means of a comparison cycle on data representative of transitions to or from each of said first and second pluralities of states.

8. A processor according to claim **7** further including means to adjust the value of said optimum transitions determined by said third computation means.

9. A processor according to claim **7** further including:

A. fourth computation means to compare data representative of candidate transitions to or from each one of a fifth plurality of states provided by said stage of said dynamic programming problem to determine the optimum one of said fifth plurality of state transitions;

B. fifth computation means to compare data representative of candidate transitions to or from each one of a sixth plurality of states provided by said stage

of said dynamic programming problem, to determine the optimum one of said sixth plurality of state transitions;

C. sixth computation means to compare data representative of the optimum ones of said state transitions of said fifth and sixth pluralities, simultaneously with comparison by said fourth and fifth computation means of data respectively representative of candidate transitions to or from seventh and eighth pluralities of states provided by said successive stage of said dynamic programming problem, to determine the optimum one of said state transitions of said seventh and eighth pluralities; and

D. means to adjust the values of said optimum transitions determined by said third and sixth computation means by one or more factors representative of variables which are constant for all of the transitions computed to arrive at the respective optimum transitions provided by said third and sixth computation means.

10. In a processor for solving dynamic programs, the combination comprising:

A. first computation means to compare data representative of candidate transitions to or from each one of a first plurality of states provided by a dynamic programming problem, to determine the optimum one of said first plurality of state transitions;

B. second computation means to compare data representative of candidate transitions to or from each one of a second plurality of states provided by said dynamic programming problem, to determine the optimum one of said second plurality of state transitions; and

C. a serial memory to store and present simultaneously to said first and second computation means from serially related locations, said data respectively associated with said transitions to or from said first and second pluralities of states, for simultaneous determination of said optimum transitions of said first and second pluralities; which serial memory is adapted to sequentially advance data between said serially related locations as said first and second computation means makes said comparisons, whereby data representative of different transitions is presented for comparison to said first and second computation means upon completion by said first and second computation means of a comparison cycle on data representative of candidate transitions to or from each of said first and second pluralities of states.

* * * * *